GENERATION OF MULTIVARIATE NORMAL SAMPLES FOR STUDYING STATISTICAL PROPERTIES OF THE EXCURSIONS OF A STOCHASTIC PROCESS ABOVE A CERTAIN LIMIT

David Scoulding

A MAJOR TECHNICAL REPORT

in the

Faculty of Engineering

Presented in Partial Fulfilment of the Requirements for

the Degree of Master of Engineering at

Concordia University

Montreal, Canada

February, 1975

Generation of multivariate normal samples for studying statistical properties of the excursions of a Stochastic Process above a certain limit

by

D. Scoulding

ABSTRACT

The purpose of this work is to generate a true random number sequence, formulate a stochastic process and study the statistical properties of its excursions for engineering applications.

Basically, a combined generator is used for generating a random number sequence. This generator is subjected to several acknowledged statistical tests, including the rigorous Kolmorogov-Smirnov test, to prove that it is a true random number generator.

This generator is then used to generate random variables, random normal variables and stochastic processes of known statistical properties. This report presents a method for the determination of the statistical properties of the excursions of such stochastic processes above a certain limit. If the statistical properties of the stochastic processes and the excursions can be related, it is expected that useful practical applications in many fields of science and engineering will result.

ACKNOWLEDGEMENTS

The author wishes to express his gratitude and deep appreciation to his supervisor Dr. T.S. Sankar for initiating the project and providing continued guidance throughout the work. Thanks are also due to Dr. K. Anantha for his valuable assistance.

TABLE OF CONTENTS

TABLE OF CONTENTS

NOMENCLATURE

$X(t), x$ - random variable

T - sample space

t - arbitrary element of T

ϵ - "belongs to"

ψ_X^2 - mean square value

μ_x - mean value

σ_X^2, VAR(X) - variance

ΔX - increment of X

T_x - time a random variable $X(t)$ lies between X and $X + \Delta X$

$p(X)$ - probability density function

$P(X)$ - probability distribution function

X_i - random number

χ^2 - chi-square statistic

r - length of a run of random numbers

$E(r), E(G)$ - expected number of runs

N - total number of random numbers

S_1, S_2, S_3 - serial lag properties

$P_1 - P_3$ - Kolmogorov-Smirnov test statistics

D - difference between sample and theoretical cumulative distribution functions

$f(x), g(x)$ - functions of x

U, U' - uniform random number in interval $(0, 1)$

$U(a, b)$ - uniform random number generator

NOMENCLATURE (Continued)

RNOR	- random normal variable generator
E	- expected value
K	- number of excursions of a stochastic process above a limit
SUMK	- total number of excursions for N stochastic processes
AVE	- average number of excursions per stochastic process
IDURN(K)	- duration of each excursion
ICDURN(K)	- number of excursions of length L
PDNDRN(K)	- probability density of duration of length I
PDSDRN(K)	- probability distribution function of duration of length I
$\bar{Y}$	- stochastic process
Y(I)	- i'th element of stochastic process
A	- covariance matrix
G	- matrix such that $GG^T=A$
α, β	- indicies
Δ	- time interval
B	- limit
NP	- number of stochastic processes

CHAPTER 1

INTRODUCTION

1.1 Object of the Technical Report

The object of this report is to take a given combined random number generator and prove that it is a true random number generator by applying several statistical tests. Then, using the generator, generate a stochastic process, study the statistical properties of the excursions of this process about a certain limit and relate these properties with those of the process.

This information can be used in engineering and other applications as given in Section 1.3.

1.2 Introduction to Some Aspects of Random Processes

A random variable is a real-valued point function X(t) defined over the sample space T of a random experiment where t is an arbitrary element of T. Such a random variable is written mathematically as $\{X(t), t \in T\}$ where the Greek letter $\in$ is read "belongs to" or "varying in". For each t in T, the observation X(t) is an observed value of a random variable.

A family of random variables $\{X(t), t \in T\}$ is called a random or stochastic process. Hence if an observed random variable is a realization of a stochastic process, then the statistical properties of the stochastic process can be inferred by applying statistical theory to this random variable.

Four main types of statistical functions are used to describe the basic properties of random data:

- Mean square values – these give an elementary description of the intensity of the data.
- Probability density functions – these give information concerning the properties of the data in the amplitude domain.
- Autocorrelation functions – these give information concerning the properties of the data in the time domain.

Power spectral densities	-	these give information concerning the properties of the data in the frequency domain

As the first two statistical functions are used in this report, they are described in more detail below.

1.2.1 Mean Square Values

The mean square value of any random data is the average of the squared values of the time history of the data. In equation form, the mean square value ψ_x^2 for a sample time history record X(t) is given by

$$\psi_x^2 = \lim_{T \to \infty} \frac{1}{T} \int_0^T X^2(t)\, dt$$

It is often desirable to think of physical data in terms of a combination of a static or time invariant component and a dynamic or fluctuating component. The static component may be described by a mean value which is simply the average of all values. In equation form, the mean value μ_x is given by

$$\mu_x = \lim_{T \to \infty} \frac{1}{T} \int_0^T X(t)\, dt$$

The dynamic component may be described by a variance which is simply the mean square value about the mean. In equation form, the variance σ_x^2 is given by

$$\sigma_x^2 = \lim_{T \to \infty} \frac{1}{T} \int_0^T \left[X(t) - \mu_x \right]^2 dt$$

The positive square root of the variance is called the standard deviation. By expanding this equation it is seen that the variance is equal to the mean square value minus the square of the mean value. That is,

$$\sigma_x^2 = \psi_x^2 - \mu_x^2$$

1.2.2 Probability Density Functions

The probability density function for random data describes the probability that the data will assume a value within some defined range at any instant of time. Consider the sample time history record of X(t) as illustrated in figure no. 1.1.

The probability that X(t) assumes a value within the range between X and $X + \Delta X$, where ΔX is an increment in X, is obtained by taking the ratio of T_x/T where T_x is the total amount of time that X(t) falls inside the range $(X, X+\Delta X)$ during an observation time T. This ratio will approach an exact probability description as T approaches infinity. In equation form,

$$\text{Prob}[X < X(t) < X + \Delta X] = \lim_{T \to \infty} \frac{T_x}{T}$$

For small ΔX a first-order probability density function $p(X)$ can be defined as follows

$$\text{Prob}[X \leq X(t) < X + \Delta X] = p(X) \cdot \Delta X$$

More precisely,

$$p(X) = \lim_{\Delta X \to 0} \frac{Prob[X \leq X(t) < X + \Delta X]}{\Delta X}$$

$$= \lim_{\Delta X \to 0} \lim_{T \to \infty} \frac{1}{T}\left(\frac{T_X}{\Delta X}\right)$$

The probability density function $p(X)$ is always a real-valued, non-negative function.

The probability that the instantaneous value X(t) is less or equal to some value X is defined by P(X), which is equal to the integral of the probability density function from minus infinity to X. This function P(X) is known as the probability distribution function.

$$P(X) = Prob[X(t) < X] = \int_{-\infty}^{X} p(\epsilon)\, d\epsilon$$

The distribution function P(X) is bounded by zero and one, since the probability of X(t) being less than $-\infty$ is zero while the probability of X(t) being less than $+\infty$ is unity i.e. a certainty. The probability that X(t) falls inside any range (X_1, X_2) is given by

$$P(X_2) - P(X_1) = Prob[X_1 < X(t) < X_2] = \int_{X_1}^{X_2} p(X)\, dX$$

In terms of the probability density [redacted] $p(X)$ the mean value of X(t) is given by

$$\bar{X} = \int_{-\infty}^{+\infty} X \cdot p(X)\, dX$$

Similiarly, the mean square value is given by

$$\psi^2_x = \int_{-\infty}^{+\infty} X^2 . p(X) . dX$$

The principal application for a probability density function measurement of physical data is to establish a probabilistic description for the instaneous values of the data.

1.3 Applications of this Report

From the above very brief description, it can be seen that the evaluation of the stochastic properties of a random process can be used to provide important information in real life applications. A few of these applications are:

(i) determination of the response of a physical process to a random exciting force.

(ii) determination of surface roughness variations from a machine operation.

(iii) establishment of maintenance routines from measurement of vibration levels of rotating machinery.

(iv) determination of outcome of experiments involving random forces in a mechanical system or turbulent velocity measurements.

(v) reliability studies.

(vi) fatigue life estimations.

Apart from obvious engineering applications, a study of the stochastic properties of a random process is useful in determining:

(i) price level variations of a commodity

(ii) population growth

(iii) commodity demand growth

(iv) medical studies etc.

1.4 Summary of Succeeding Chapters

In this report, the generation of a random sequence is described in Chapter 2 using a combined generator. In Chapter 3, statistical tests are given to show that the number sequence so generated is a true random number sequence. The generation of a random variable from this sequence is described in Chapter 4. The Kolmorogov-Smirnov test, the most severe of all statistical tests of randomness, is used to show that the variable, thus generated, is indeed random. The development of a Stochastic Process and the evaluation of its statistical properties above a certain limit is covered in Chapter 5.

The symbols used in this report are defined in the nomenclature and are also described in the text when they appear for the first time.

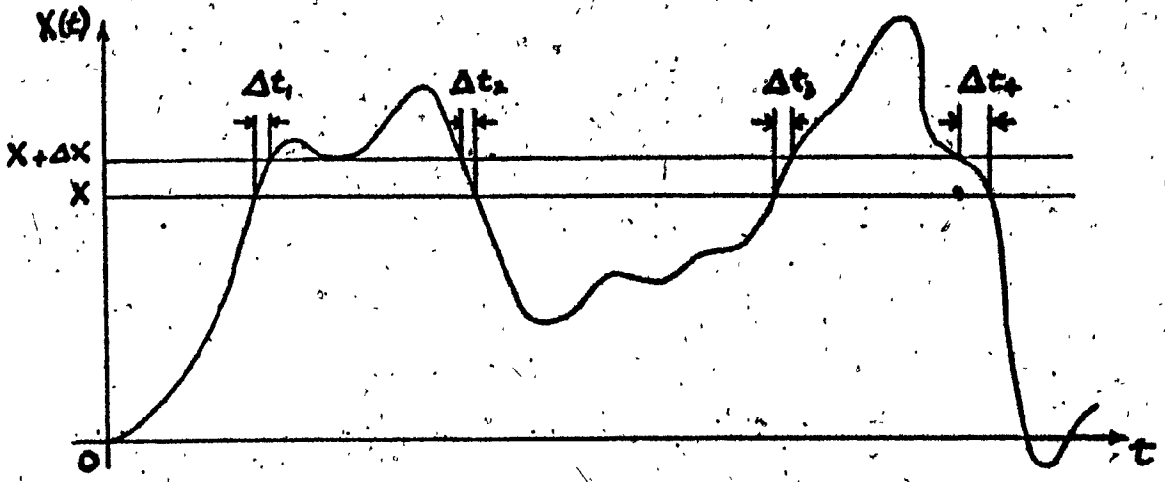


Fig. 1.1 Sample Time History Record of X(t)

CHAPTER 2

GENERATION OF A RANDOM NUMBER SEQUENCE

2.1 Introduction

A random number generator is basically a set of computer instructions which scramble the digits of an integer to produce a new integer. Here "scramble" means to mix and combine by means of simple computer operations. The idea is to produce a sequence of integers which, in spite of being produced by a fixed procedure, will serve as random variables in computer simulations.

The three types of random number generators that have dominated both practical use and theoretical discussion in the literature are:

Mid-Square Generator

Given an 8-digit decimal integer, square to get 16 digits, keep the middle 8 digits:

current I = 45086273

I = 2032772013030529

new I = 77201303

Congruential Generator

Given an 8-digit integer, multiply by a constant, say 7654321, then keep the last 8 digits:

current I = 45086273

I x 7654321 = 345104806233633

new I = 06233633

Shift Register Generator

Given an 8-digit integer, shift right 3, do a no-carry add, shift left 4, then do another no-carry add:

current I =	45086273	
	00045086	shift right 3
	45021259	no-carry add
	12590000	shift left 4
new I =	57511259	no-carry add

Each of these examples produces a sequence of integers that seem to jump around haphazardly within the allowable range (word size) of the computer. There are two basic problems: (1) to make sure that the generator does not begin to repeat itself too soon, and (2) to somehow test the output for randomness.

The mid-square generator is now considered obsolete. Congruential generators, by far the most commonly used method for the past 25 years, have been the subject of numerous papers. Recent discoveries have shown that congruential generators have certain regularities. Hence they are not as satisfactory as they were thought to be. Shift register generators, used for many years in communication theory, coding theory and cryptography, have been recently heralded as the best alternative to congruential generators. However, shift register generators have regularities that appear to be more serious, though not as easy to establish, as those of the congruential generators.

However it has recently been found (8), (9) that combining the congruential and shift register generators in various ways appears to produce satisfactory sequences. Combined generators have passed extensive tests for randomness as well as resisted attempts to display regularities of the type found when congruential or shift register generators are used alone.

2.2 Generation of a Random Number Sequence

For the purposes of this report, uniform random numbers in the interval (0, 1) are generated using a combined generator. Computer program no. 1 (RAND) in Appendix B shows the actual generator used. The program was given to the author by Dr. Anantha.

The function UNI (DUMMY) gives a uniform random number. The argument, DUMMY, of the function is any integer. In this program, DUMMY is arbitrarily set to 1234567. The integers 69128 and 69069 are known to give a long sequence of random numbers without any discernible regularities.

The first one hundred numbers thus generated are listed at the end of the program.

CHAPTER 3

STATISTICAL TESTS

3.1 Introduction

In order to generate a Stochastic Process using a random number sequence, it is necessary to show that the random number sequence used is truly random. For this purpose several statistical tests used for various congruential pseudo-random number generators [1,2,3] were performed.

Programs were developed to apply the statistical tests. These are described and the results listed in the following sections. These tests use the Chi-square test of significance which is described in Appendix A.

The Kolmorogov-Smirnov test, presently regarded in the literature as the most severe of all the tests, is also performed on the generated sequence of random numbers. A concluding section summarizes the results.

3.2 Test for Uniformity

3.2.1 Method

In this test, the unit interval (0,1) was divided into 10 equal intervals (i.e. 0.0 to 0.099, 0.10 to 0.199, etc.). A sequence of 1000 uniform random numbers in the interval (0,1) was generated using the method described in Chapter 2. This is used in all the following tests in this chapter. The occurrences of these 1000 numbers in each of the intervals were counted. The results are given in the following sub-section.

The chi-square statistic $\left(\chi^2 = \sum \frac{(O-E)^2}{E}\right)$ as given in Appendix A, section 2 was computed. The number of degrees of freedom was determined as given in Appendix A, section 3. Then using Table 1, in Appendix A, the probability (P) that there is no significant difference between the distribution of the random number sequence generated and a true random number sequence was determined. The criteria given in section 4 of Appendix A was then applied to check that the generated number sequence successfully passed this test for randomness.

This technique is repeated in the following statistical tests in this chapter.

Details of the computer program for the test for uniformity are shown in program no. 2 (UNIG).

3.2.2 Results

Interval	No. of Occurrences	Expected no. of Occurrences
$0.0 \leq X_i < 0.1$	86	100
$0.1 \leq X_i < 0.2$	119	100
$0.2 \leq X_i < 0.3$	93	100
$0.3 \leq X_i < 0.4$	105	100
$0.4 \leq X_i < 0.5$	89	100
$0.5 \leq X_i < 0.6$	100	100
$0.6 \leq X_i < 0.7$	111	100

Interval	No. of Occurrences	Expected no. of Occurrences
$0.7 \leq X_i < 0.8$	108	100
$0.8 \leq X_i < 0.9$	90	100
$0.9 \leq X_i < 1.0$	99	100

Where X_i is a random number

χ^2 = 10.38

Number of degrees of freedom = 9

Probability (P) = 32%

3.3 Test for Runs Above and Below the Median

3.3.1 Method

In this test, the number of runs of consecutive random numbers above or below the median were calculated and compared with expected values.

Given a sequence of numbers $X_1 \ldots\ldots\ldots\ldots X_N$, the subsequence $X_{i-1} \ldots\ldots\ldots\ldots X_{i+r}$ is said to form a run of length r above the median if

$$X_{i-1} < \tfrac{1}{2},\ X_i \geq \tfrac{1}{2}, \ldots\ldots X_{i+r-1} \geq \tfrac{1}{2},\ X_{i+r} < \tfrac{1}{2}.$$

Similarly a run of length r below the median is defined by

$$X_{i-1} > \tfrac{1}{2},\ X_i \leq \tfrac{1}{2}, \ldots\ldots X_{i+r-1} \leq \tfrac{1}{2},\ X_{i+r} > \tfrac{1}{2}.$$

End runs are defined by

$i = 1$ or $i+r = N+1$.

The expected number of runs of length r, not distinguishing between runs above and below, is approximately given by

$$E(r) = (N - r + 3)2^{-(r+1)}$$

and the expected number of length r or greater by

$$E'(r) = (N - r + 2)2^{-r}$$

where N is the total of numbers in the sequence.

The derivation of these formulae is given in (5).

Again a sequence of 1000 uniform random numbers in the interval (0,1) was used. The number of runs of consecutive numbers of lengths 1 to 9 inclusive and 10 or greater above or below the median of 0.5 were determined.

From these results and the expected number of runs as determined from the above formulae, the chi-square goodness of fit test was performed. Details of this method are shown in program no. 3 (AEMD).

3.3.2 Results

Length of run	Expected No. (from above formulae)	Actual No.
1	250	249
2	125	127
3	63	68
4	31	28
5	16	12
6	8	10
7	4	1
8	2	3
9	1	1
10 or greater	2	2

χ^2 = 5.04

Number of degrees of freedom = 9

Probability (P) = 83%

3.4 Test for Runs Up and Down

3.4.1 Method

In this test the number of runs of random numbers in ascending and descending orders were determined and compared with expected values.

Given a sequence of numbers $X_1 \ldots\ldots\ldots\ldots X_N$, the subsequence $X_{i-1}, \ldots\ldots\ldots, X_{i+r+1}$ is said to form a run of length r "up" if $X_{i-1} > X_i$,

$X_i < X_{i+1} < X_{i+2} \ldots\ldots\ldots\ldots < X_{i+r} > X_{i+r+1}$

Similarly a run of length r "down" is defined by

$X_{i-1} < X_i > X_{i+1} > X_{i+2} \ldots\ldots > X_{i+r} < X_{i+r+1}$

End runs are defined by $i = 1$ or $i = N-r$.

The expected number of runs of length r not distinguishing between runs up and down, is approximately given by

$$E(r) = 2\{N(r^2+3r+1) - (r^3+3r^2-r-4)\}/(r+3)!$$

and the expected number of length r or greater by

$$E'(r) = 2\{N(r+1) - (r^2+r-1)\}/(r+2)!$$

where N is the total of numbers in the sequence.

The derivation of these formulae is given in (5).

Again a sequence of 1000 uniform random numbers in the interval (0,1) was used. The number of runs of length 1 to 4 inclusive and 5 or greater were determined. From these results and the expected number of runs as determined from the above formulae, the chi-square goodness of fit test was performed. Details of this method are shown in program no. 4 (UPDO).

3.4.2 Results

Length of run	Expected no. (from above formulae)	Actual no.
1	416.8	406
2	183.1	196
3	52.7	57
4	10.5	5
5 or greater	2.4	2

χ^2 = 4.46

Number of degrees of freedom = 4

Probability (P) = 35.4%

3.5 Test for Serial Lag

3.5.1 Method

In this test, the unit square was divided into 100 equal cells. Pairs of random numbers were taken as the co-ordinates of a point within the unit square. The number of points within each cell were counted and compared with the expected values.

From (6), to test the serial lag properties of a sequence (for lag q, $1 \leq q \leq 6$), a sample of 1000 plus q random numbers was used. The interval (0,1) was divided up into 10 equal disjoint intervals. Let f_i be the number of numbers X_r in the range $\left(\frac{i-1}{10}, \frac{i}{10}\right)$, and $q_{i,j}(q)$ be the number of ordered pairs (X_r, X_{r+q}) such that X_r contributes to f_i and X_{r+q} to f_j.

Let $$S_1^2 = \frac{1}{100} \sum_{i=1}^{10} (f_i - 100)^2$$

and $$S_2^2(q) = \frac{1}{10} \sum_{i=1}^{10} \sum_{j=1}^{10} (q_{ij}(q) - 10)^2$$

Then $S_3^2 = S_2^2(q) - S_1^2$ is asymptotically χ^2_{90} for a truly random sequence (6). S_3^2 was calculated and compared with the distribution of χ^2_{90}. This test was repeated for co-ordinate points of Xi, Xi + 1 etc., Xi, Xi + 2 etc. ....., Xi, Xi + q where q = 6 to give six sets of results. Details of this method are shown in program no. 5 (SERIAL).

3.5.2 Results

A study of the matrix printouts in program no. 5 for lags of 1 to 6 inclusive shows a reasonably even distribution of the pairs of number for each lag. In this program, the χ^2 values were computed incorrectly due to a programming error. The necessary corrections to lines 44, 46, and 47 of the program are shown in program no. 5 in Appendix B.

3.6 Test for Sum of N Numbers

3.6.1 Method

In this test, $P\left\{(u_1 + u_2 + \ldots\ldots + u_n)/n\right\}$ is determined using 10 equal intervals in the interval (0,1), where P is the distribution function of the sum of n random numbers. This distribution is compared with that expected of a true random number sequence to prove the generated sequence as being random.

The expected distribution is determined as follows:

Let $z = U_1$ where U_1 is a uniform random number

$$\therefore p(z) = 0, \quad z<0$$
$$= 1, \quad 0<z<1$$
$$= 0, \quad z>1$$

where $p(z)$ is the probability density function and

$$P(z) = 0, \quad z<0$$
$$= z, \quad 0<z<1$$
$$= 1, \quad z>1$$

where $P(z)$ is the probability distribution function.

Now for $z = U_1 + U_2$ where U_1 & U_2 are uniform random numbers

then

$$p(z) = 0, \quad z<0$$
$$= z, \quad 0<z\leq 1$$
$$= 2-z, \quad 1<z<2$$
$$= 0, \quad z>2$$

and

$$P(z) = 0, \quad z<0$$
$$= z^2/2, \quad 0<z<1$$
$$= -z^2/2+2z-1, \quad 1<z<2$$
$$= 1 \quad z>2$$

from

$$P(z) = \int p(z)\cdot dz$$

This is repeated to determine $p(z)$ & $P(z)$ for $z = U_1+U_2+U_3$, $U_1+U_2+U_3+U_4$, & $U_1+U_2+U_3+U_4+U_5$.

A sequence of 1000 uniform random numbers in the interval (0,1) was generated using the method described in Chapter 2. The

distribution function for n = 2, 3, 4, & 5 was determined and compared with the expected value. The chi-square goodness of fit test performed. Details of this method are shown in program no. 6 (SUMN).

3.6.2 Results

Number of degrees of freedom = 9.

N	χ^2	probability (P)
2	8.05	53%
3	8.83	46%
4	3.69	93%
5	3.82	92%

3.7 Poker Test

3.7.1 Method

For the generated sequence of 1000 numbers, the first digits of 5 consecutive numbers in the sequence were examined and classified according to the type of poker hand to which they corresponded, as shown in the following table:

Event	Probability
bust	0.3024
one pair	0.5040
two pairs	0.1080
three of a kind	0.0720
full house	0.0090
four or more of a kind	0.0046

The probabilities of the different arrangements of the digits are tabulated in (5).

A standard Chi-square test of goodness of fit was performed using the expected number and actual number of each type of hand. Details of this method are shown in program no. 7 (PQKF).

3.7.2 Results

Event	Expected No.	Actual No.
bust	60.5	66
one pair	100.8	96
two pairs	21.6	20
three of a kind	14.4	15
full house	1.8	3
four or more of a kind	0.9	0

χ^2 = 2.596

Number of degrees of freedom = 5

Probability (P) = 76%

3.8 Plot of Random Number Sequence on X - Y Plane

3.8.1 Method

Consecutive pairs of random numbers in the sequence were taken to form co-ordinates of a sequence of points. The co-ordinates were scaled to give a print-out of points on a 54 x 90 scale. This scale was chosen to fit on a page of computer printout. This was repeated for N equals 500 to 1000 points in

100 point steps. Details of this method and the resultant printouts are shown in program no. 8 (PLOT).

3.8.2 Results

As the sequence was never repeated for each value of N, six independent plots were obtained. As these plots do not contain any discernible regularities, the generated random number sequence can be said to pass this visual test.

3.9 The Kolmorogov-Smirnov Test for a Uniform Distribution (KS TEST)

3.9.1 Method

In the literature, the Kolmorogov-Smirnov test (10), (11), is regarded as the most severe of all statistical tests of a random number sequence. In this test, a sample of "N" random numbers is arranged in order of ascending magnitude. The sample cumulative distribution function is constructed as a step function rising from zero to one.

The difference D between the sample and the theortetical cumulative distribution functions is determined. This is a random variable itself because repetitions of its computations will vary among randomly selected sequences of N numbers.

The computer program used for this test was given to the author by Dr. Anantha. Details are given in program no. 9 (KOLSMI). The subroutine ASORT arranged the generated sequence of random

numbers into ascending values of magnitude. The sample cumulative distribution function is determined and compared with the theoretical in the subroutine KSTEST which gives the following five results:

$P_1 = P(D+)$ - distribution function of positive differences

$P_2 = P(D-)$ - distribution function of negative differences

$P_3 = P(MAX(D+,D-))$ - distribution function of the maximum absolute difference, known as the K-S statistic

P_4 - area between distribution functions (Cramer - Von Mises)

P_5 - weighted Cramer - Von Mises

If the values of P_1 to P_5 inclusive are greater than 0.05 but less than 0.95, then the generated sequence of random numbers is said to have a uniform distribution.

3.9.2 Results

P_1 = 0.2517

P_2 = 0.5359

P_3 = 0.1625

P_4 = 0.1318

P_5 = 0.1847

Thus, it can be concluded that the random number sequence has a uniform distribution.

3.10 Summary of Statistical Tests

It can be reasonably concluded that the given random number generator is a true random number generator since it passed all the statistical tests including the rigorous Kolmorogov-Smirnov test.

This generator can therefore be used to simulate a random variable and/or stochastic process.

As the Kolmorogov-Smirnov test is regarded as the most severe test for a randomness, it is the only test used in subsequent work.

CHAPTER 4

GENERATION OF A RANDOM VARIABLE, A NORMAL RANDOM VARIABLE, WHITE NOISE AND A STOCHASTIC PROCESS

4.1 Generation of a Random Variable

4.1.1 Method

This section briefly describes how a random variable can be generated using the random number sequence. The Kolmorogov-Smirnov test is applied to two such simple variables.

Let $f(x) = e^{-x} \quad x \geq 0$

$= 0 \quad x < 0$

$$\therefore P(a) = \int_0^a f(x).dx$$

$$= \int_0^a e^{-x}.dx$$

$$= \left[-e^{-x}\right]_0^a$$

$$= 1 - e^{-a}$$

By setting, U = P (a), where U is a uniform random variable (0,1), we can write

$$1 - e^{-x} = U$$

$$\text{or } e^{-x} = 1 - U$$

$$\therefore x = -\log (1 - U)$$

Now U is uniform (0,1)

Hence 1 - U is uniform (0,1)

Therefore $x = -\log U'$

where U' is also uniform (0,1)

4.1.2 Results

Program no. 10 (RANVAR 1) generates $x = -\log (1 - U)$ and this generated sequence passed the KS test. Program no. 11 (RANVAR 2) generates $x = -\log U'$ and this sequence also passed the KS test.

4.2 Generation of a Normal Random Variable

4.2.1 Technique

Let $f(x)$ be a known probability density function and $g(x)$ be a probability density function such that $g(x) > f(x)$ for all x as shown in figure 4.1. Let

$$g_1(x) = \frac{g(x)}{\text{area under } g(x) \text{ curve}}, \quad c = \frac{1}{\text{area under } g(x) \text{ curve}}$$

$$\therefore g_1(x) = c \cdot g(x)$$

$$\therefore g_1(x) > c \cdot f(x)$$

From this inequality, the following rejection technique is developed to generate a random variable X with a probability density function f(x).

Step 1. Generate the random variable X with density $g_1(x)$.

Step 2. Let X_1 be a sample. If $g_1(X_1) \cdot \text{UNIF}(0) < c \cdot f(x)$ go to step 4.

Step 3. Go to step 1.

Step 4. X_1 has density f(x) since $c \cdot f(x)$ can be scaled to become f(x).

4.2.2 Method

The Cauchy density function has a shape similar to a normal density function, see figure 4.2.

The Cauchy density function is given by:

$$f(x) = \frac{1}{\pi(1+x^2)}$$

$$\text{Now, } P(x) = \int_{-\infty}^{x} f(t)\,dt$$

$$\therefore P(x) = \frac{1}{\pi}\int_{-\infty}^{x} \frac{1}{1+t^2}\,dt$$

$$= \frac{1}{\pi}\left[\tan^{-1} t\right]_{0}^{x}$$

$$= \frac{1}{2} + \frac{\tan^{-1}(x)}{\pi}$$

$$\text{Let } P(x) = U,$$

$$\therefore x = \tan \pi\left(U - \tfrac{1}{2}\right)$$

Hence the random variable x can be generated using the random number sequence U to fit the Cauchy distribution.

Now the normal density function is given by:

$$f(x) = \frac{e^{\frac{-x^2}{2}}}{\sqrt{2\pi}}$$

and scaling factor, $c = \sqrt{\frac{2}{\pi}}$

Program no. 12, (RANNOR) uses the above method to generate a random variable to which the KS test is applied.

4.2.3 Results

From the results of this program:

P_1 = 0.11

P_2 = 0.71

P_3 = 0.44

P_4 = 0.65

P_5 = 0.53

As these values are within the limits of 0.05 to 0.95, the random variable generated has a normal distribution.

4.2.4 Further Programs

The above method used a self-generated program to develop a random variable with normal distribution. There are several other methods in the literature (8). For completeness two of these were selected and subjected to same test. The computer program was also modified to permit the test to be repeated 10 times for each set of 1000 numbers.

Program no. 13 is based on the use of polar co-ordinates. All runs except the ninth were acceptable. This shows that any random variable generated this way should be thoroughly tested before use.

Program no. 14 used a one-line generator. Two runs were rejected, again showing that testing before use is essential.

It is not the intention of this report to explain these programs any further. However, it should be noted that when a large number of tests are performed (10 in the case of the above programs), there is a statistical probability that individual tests will fail even if the sequence of numbers is truly random. This probability can be evaluated as shown in (10).

4.3 Use of a Random Number Sequence in Determining Probabilities

4.3.1 Introduction

This section gives two simple examples of how a random number sequence can be used to determine probabilities,

4.3.2 Throwing of a Pair of Dice

4.3.2.1 Method

If a pair of dice are thrown together, the probability of the sum of the dice being, 2, 3 ..., to 12 is given in the next section. If pairs of numbers in a random number sequence are used to simulate pairs of dice using these formulae:

Dice 1 Face Value = $6X_i + 1$ $\quad 0 \leq X_i < 1$

Dice 2 Face Value = $6X_{i+1} + 1$ $\quad 0 \leq X_{i+1} < 1$

then these can be added and, if repeated enough times, should give probabilities close to the theoretical.

Program No. 15 (DICE) uses this method by taking 1000 consecutive pairs of the random number sequence.

4.3.2.2 Results

Face Value of Die	Theoretical Probability	Probability from 1000 throws
2	0.0277	0.0330
3	0.0554	0.0470
4	0.0831	0.0860
5	0.1108	0.1130
6	0.1385	0.1390
7	0.1662	0.1700
8	0.1385	0.1450
9	0.1108	0.0970
10	0.0831	0.0760
11	0.0554	0.0570
12	0.0277	0.0370

The results show that the random number simulator produced values close to the expected values.

4.3.3 Determination of Probability of an Event From a Given 3 x 3 Probability Matrix

4.3.3.1 Method

Program no. 16 (MATQ2) shows how starting with a given probability matrix, a random number sequence can be used to determine the probability of the individual events. The following matrix in addition to a 1000 random number sequence was used:

.4	.3	.3
.2	.3	.5
.4	.2	.4

4.3.3.2 Results

Expected Probability		Probability from program
$\frac{.4 + .2 + .4}{3.}$	= 0.333	0.332
$\frac{.3 + .3 + .2}{3.}$	= 0.267	0.268
$\frac{.3 + .5 + .4}{3.}$	= 0.400	0.400

The results are very close to those expected.

4.4 Programs to Plot White Noise and a Stochastic Process

4.4.1 Generation of White Noise

A time series, comprising of random variables that are independently distributed according to the same cumulative distribution function, is known as white noise. A property of a random number generator is that it produce uniform white noise - a sequence of independent random variables from the same probability density function. A white noise process might alternatively be thought of as a random sample of infinite size from a particular density function.

In program no. 17 (PLRNOR), the one-line generator was used to generate a normal variate as previouly discussed in section 4.2.4. The computer time taken, using this generator, is much shorter than that taken using the techniques in sections 4.2.1 & 4.2.2. For this reason, the one-line generator is used in all subsequent work.

One hundred random normal variables were generated and scanned for maximum and minimum values to obtain a scaling factor. The one hundred values were then plotted on one computer printout sheet using a uniform time interval between each variable. This was repeated for 10 sets of one hundred values. The resultant 10 processes should represent white noise by definition.

Program no. 17 (PLRNOR) shows the program and results. A visual study of the results shows a random distribution of points as expected for white noise.

4.4.2 Generation of a Stochastic Process

A time series defined either by an infinite collection of random variables $\{X(t),\ t = 1, 2, 3 \ldots\ldots\ldots\}$ or a complete collection of realizations $[X(1), X(2), \ldots \ldots X(T), X(T+1), \ldots\ldots]$ is called a stochastic process.

In program no. 18 (STOPRO), the one-line generator was used to generate a normal variate. This was then used to generate a Stochastic process as follows:

Let $x_1, x_2, \ldots\ldots\ldots\ldots x_N$ be consecutive values of a stochastic process sampled at equal time intervals with common variance σ^2 and common mean 0.

Let RNOR(0) be a random normal variable. Therefore to obtain a normal with mean μ and variance σ^2, let

$$X = \mu + \sigma\, RNOR(0)$$

$$\therefore E(X) = E(\mu) + \sigma E(RNOR(0))$$

where E is the expected value

$$\therefore E(X) = \mu$$

and

$$VAR(X) = E((X-\mu)^2)$$

$$= E(\sigma^2 RNOR^2(0))$$

$$= \sigma^2$$

Therefore, by setting $x_i = \sigma\, RNOR(0)$ a stochastic process of mean 0 and variance σ^2 is obtained.

In program no. 18, σ was chosen to be 0.1 and consecutive points of the stochastic process were formed using the expression $X_i = X_{i-1} + \sigma$ RNOR(0) and a uniform time interval between each point. As for the white noise program, ten sets of one hundred values were obtained and each plotted on one computer printout sheet. Program no. 18 (STOPRO) shows the program and plots. A study of the plots show the random behaviour of the Stochastic Processes as expected.

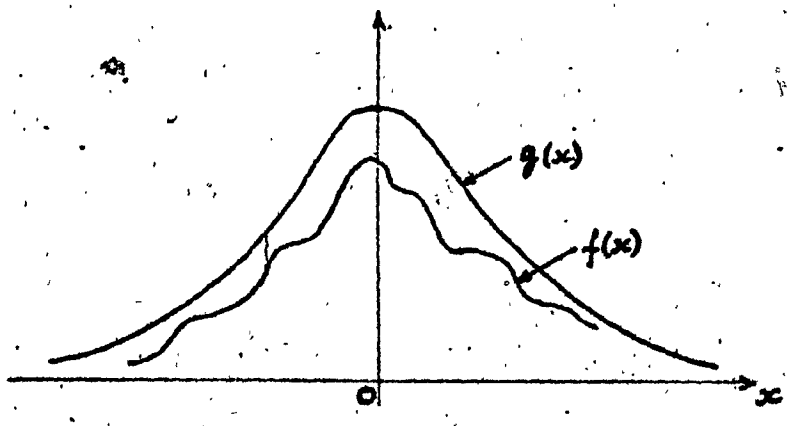


Figure 4.1 Probability Density Functions

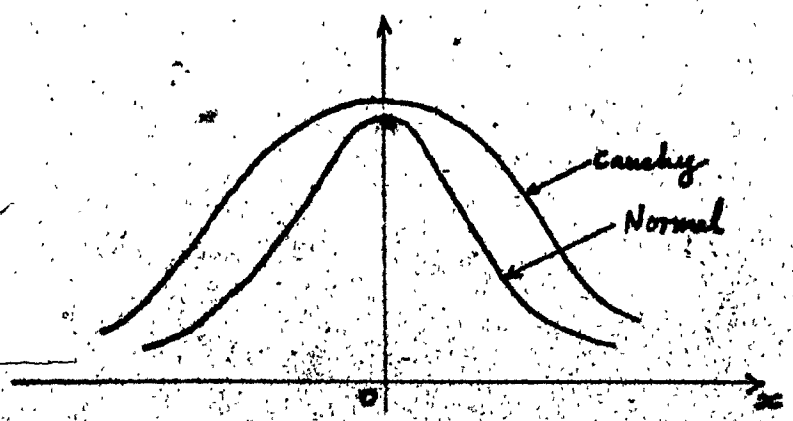


Figure 4.2 Cauchy and Normal Density Functions

CHAPTER 5

PROGRAMS TO DETERMINE THE EXCURSION PROPERTIES OF A STOCHASTIC PROCESS

5.1 Determination of the Number of Excursions of a Stochastic Process Above a Certain Limit

5.1.1 Method

In this program, a hundred consecutive values of a stochastic process were generated. A subroutine was developed to count the number of times this process crossed a certain limit and to calculate the length of each excursion above this limit. Program no. 19 (CRODUR) was developed for this purpose. The above process was repeated for one hundred different processes and the average number of crossings was determined. This was done for seven different limits.

In the program:

K	equals number of excursions above the given limit for each stochastic process of 100 values
SUMK	equals total number of excursions for the 100 stochastic processes
AVE	equals average number of excursions per stochastic process
IDURN (K)	equals the duration of each excursion

5.1.2 Results

Program no. 19 gave the following results:

Limit	Average Number of Excursions
.2	2.41
.25	2.15
.3	2.17
.35	1.96
.4	2.10
.45	1.67
.5	1.62

In spite of the fact that there are more crossings of the limit at .3 than at .25 and at .4 than at .35, the results are valid as shown in figure 5.1.

From this figure:

No. of crossings of .25 = 2

No. of crossings of .3 = 4

This behaviour can also be seen in the plot obtained in program no. 23 (STOCHY).

5.2 Determination of the Duration of Excursions of Stochastic Process Above a Certain Limit

5.2.1 Method

Program no. 20 (DURATN) details the method and results.

This program is an extension of the previous one, CRODUR. The duration of each excursion is determined. Then for each stochastic process of one hundred uniform time intervals, the number of excursions of each duration, i.e. 1 to 100, is determined. This is repeated one hundred times. Then the number of excursions of each duration is divided by the total number of excursions to determine, given an excursion, the probability density of the duration of that excursion. The probability distribution function is also determined.

In the program:

ICDURN(L)	=	number of excursions of duration L.
PDNDRN(I)	=	probability density of duration of length I.
PDSDRN(I)	=	probability distribution function of duration of length I.

5.2.2 Results

The results show that the probability of the duration being 7 time intervals or less is 44%, and 17 or less is 61%. Hence most durations are of a short time interval which is to be expected. However some processes stay above the limit for long time intervals. This is shown by 5% of the excursions having lengths of 79 time intervals or greater. In fact these processes

most probably went above the limit and never fell below it before the set 100 values had been reached.

These results are very much in line with what one would expect from a visual inspection of the graphical representations given by program no. 18 (STOPRO).

5.3 Generation of a Stochastic Process of Known Statistical Properties

5.3.1 Method

A stochastic process of known statistical properties can be generated using the method developed by Marsaglia (7). This method involves the linear transformation of the co-variance matrix as follows:

$$Y(I) = A \cdot X(I)$$

where Y is an n x 1 vector

X is an n x 1 vector representation of a random normal variable

A is a given n x n matrix

A is formed in accordance with $A_{i,j} = e^{-\beta(i-j)}$ for $i \geq j$ and $A_{i,j} = 0,\ i < j$. A is known as the co-variance matrix and the process is called a Markov process. Processes formed using $e^{-\beta(i-j)^2}$ are non-Markovian.

Hence matrix A is represented by:

$$A = \begin{bmatrix} 1 & e^{-\beta} & e^{-\beta^2} & \cdots & e^{-\beta^{n-1}} \\ & 1 & e^{-\beta} & & \vdots \\ & & 1 & \ddots & \vdots \\ & & \mathbf{0} & \ddots & \vdots \\ & & & & 1 \end{bmatrix}$$

Now $A_{i,j} = e^{-\beta(i-j)}$

$= r^{i-j}$ where $r = e^{-\beta}$

let $p = (1-r^2)^{1/2}$

Now $Y(I) = A.X(I)$

$\therefore y_1 = x_1$

$y_2 = r x_1 + p x_2$

$y_3 = r^2 x_1 + p(r x_2 + x_3)$

$y_4 = r^3 x_1 + p(r^2 x_2 + r x_3 + x_4)$

$\vdots$

$y_n = r^{n-1} x_1 + p(r^{n-2} x_2 + \ldots\ldots\ldots + x_n)$

Now $y_3 = r^2 x_1 + p(r x_2 + x_3)$

$= r(r x_1 + p x_2) + p x_3$

$= r \cdot y_2 + p \cdot x_3$

In general, this can be written as

$$y_i = \sum_{j=1}^{i} G_{i,j}\, x_j$$

where $G = \begin{bmatrix} 1 & & & \mathbf{0} \\ r & p & & \\ r^2 & rp & p & \\ r^3 & r^2 p & rp & p \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$

and it can be shown that $GG^T = A$. Hence if A is given, G can be readily determined. Further Y(I) can be calculated using

$$Y_i = \sum_{j=1}^{i} G_{i,j} x_j$$

This technique is used in program no. 21 (STPROY) where the subroutine DECOMP is used to determine G for any given A.

In this program the first one hundred values of Y(I) are determined given A and X(I) equals RNOR(0). Hence A is a 100 x 100 matrix. The components are listed in consecutive order as this reduces the memory storage from 10,000 to 5050.

5.3.2 Results

Program 21 printed out the one hundred values of Y. As a graphical representation gives a better idea of the result, this is done in another program, see Section 5.4.

5.3.3 Check

The program used for determining G such that $GG^T = A$, where A is the covariance matrix, had to be checked.

This was done in program no. 22 (GMAT1) where A was the given 4 x 4 matrix:

$$A = \begin{bmatrix} 4 & 1 & 1 & 1 \\ 1 & 5 & 1 & 1 \\ 1 & 1 & 6 & 1 \\ 1 & 1 & 1 & 7 \end{bmatrix}$$

G was determined, the product GG^T calculated and compared with A. As the results of the program show, G was correctly determined.

5.4 Program to Give Collective Results for Printout of Values, Plot, Number of, Probability Density of and Probability Function of Excursions of a Stochastic Process

5.4.1 Introduction

The last two programs that were developed were to determine the statistics of the excursions of a stochastic process above a certain limit. These programs are given in this and the following sections. Due to time limitations it was not possible to develop them fully. It is the intention of this major technical report to present these programs along with some initial results. Further work is required before a detailed interpretation of the results can be presented.

5.4.2 Method

This program essentially combines several previous programs so that the results may be presented together. In program no. 23 (STOCHY), the Stochastic Process, Y, is formed using Y(I) = A . X(I) as in section 5.3. The printout for the first 100

values is given along with a plot. The program was expanded to give not only the number crossings above the limit, the probability density and distribution function but also similar data for the crossings below the limit.

This program was further extended to permit the evaluation of the following:

Root mean square of the excursions $\psi(p)$

Mean of the excursions $\mu(p)$

Standard deviation of the excursions $\sigma(p)$

These are determined in the following manner as shown in figure 5.2.

From this figure, $\tau_1 = i_2 - i_1$

$T_1 = i_3 - i_2$

$\tau_2 = i_4 - i_3$, etc.

The root mean square, $\psi(p)$, of the excursions is given by:

$$\psi(p) = \int_0^t \tau^2 \cdot p(\tau) \cdot d\tau$$

Where $\tau = i.\delta$ and

δ = time interval

i = no. of time intervals

$$\therefore \psi(p) = \delta^2 \sum_{i=1}^{100} i^2 \cdot p(i)$$

Mean, $\mu(p)$, of the excursions is given by:

$$\mu(p) = \int_0^t \tau_i \, p(\tau) \, d\tau$$

$$= \delta \sum_{i=1}^{100} i \, p(i)$$

Standard deviation,

$$\sigma(p) = \left\{ \psi^2(p) - \mu^2(p) \right\}^{1/2}$$

In program no. 24 (STOCHY), these values were determined for the limit B = 0.4 and in program no. 25 the program was repeated for limit B = 0.0. Both programs produced the statistics for excursions above and below the limit.

These programs were also repeated for β = 0.05 and β = 0.005 where β is the constant in the formation of the matrix A in accordance with $A_{i,j} = e^{-\beta(i-j)}$ as in section 5.3.1. In both cases the computer rejected the formation of the G matrix as the term SM became small negative values which is not permissible. This problem was not investigated any further.

5.4.3 Results

5.4.3.1 Program no. 23

The program was checked as follows:

Length of crossing above limit	Number of Crossings	Length X Number
1	1	1
2	3	6
3	3	9
5	1	5
8	1	8
		29

Length of crossing below limit	Number of Crossings	Length X Number
2	2	4
3	1	3
4	2	8
5	1	5
10	1	10
13	2	26
15	1	15
		71

Total = 29 + 71 = 100, as expected.

This can also be checked from the plot of the stochastic process. As the program was run for one set of one hundred values of Y, the results are not sufficient to be able to draw any firm conclusions.

5.4.3.2 Programs nos. 24 & 25

	B = 0.4	B = 0.0
No. of crossings above	9	8
μ (p)	0.03222	0.05625
var. (p)	0.00143	0.00428
σ (p)	0.01987	0.03351
No. of crossings below	10	9
μ (q)	0.07100	0.06222
var. (q)	0.00737	0.00540
σ (q)	0.04826	0.03909

5.5 Program to Determine the Statistics of the Excursions of a Stochastic Process Above a Certain Limit

5.5.1 Method

The previous program determined the statistics of the excursions for one set of one hundred values of a stochastic process. For more realistic results, it is necessary to determine the statistics for more than one set. Program no. 26 (CROSSY) evaluates the statistics for an ensemble of one hundred sets of one hundred values each.

5.5.2 Results

The results obtained are shown in the program. The values for the probability density and distribution function of the excursions below the limit are incorrect due to an error in the subroutine cross which gave 792 crossings which, in fact, a check showed that there were only 742.

This error was corrected in program no. 27 which also generalizes the number (NN) of sets of one hundred values used. Unfortunately, for the runs made with this program, the values of β (equals 0.05 and 0.005) were such that the matrix G could never be formed for the reason given in section 5.4. Hence no results exist for this program.

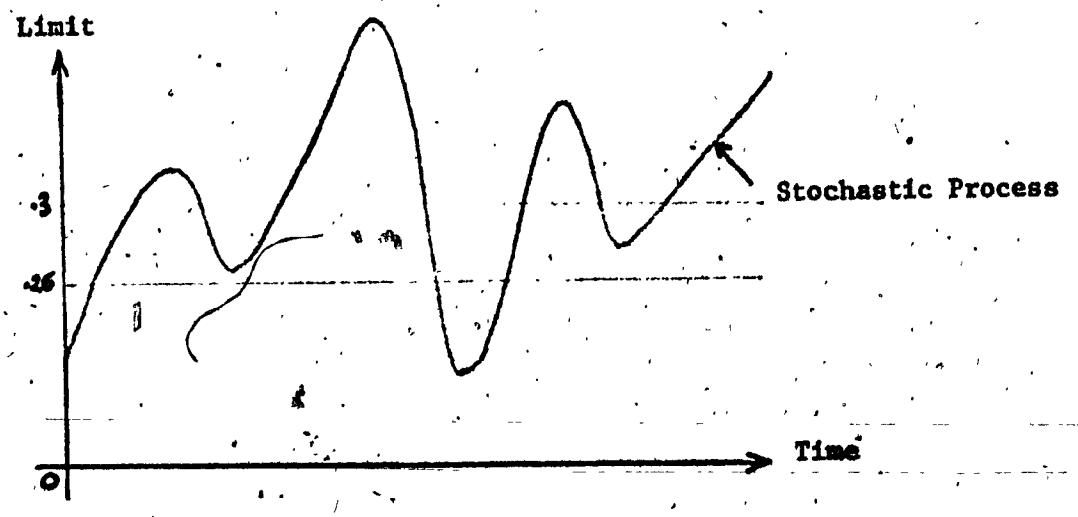


Figure 5.1 Behaviour of a Stochastic Process Crossing Limits

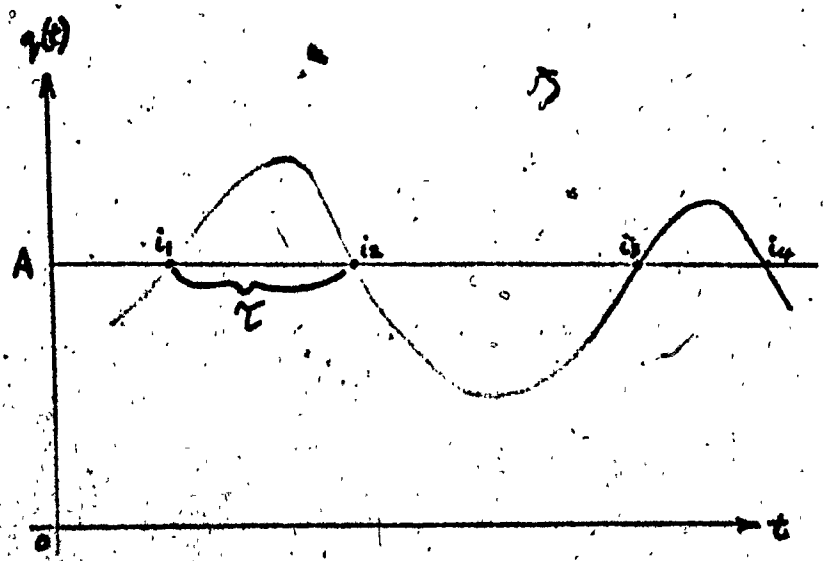


Figure 5.2 Determination of Statistical Properties of Excursions

CHAPTER 6

DISCUSSION OF RESULTS

Chapter 3 on the statistical tests on the random number generator did show that it passed all known tests. Therefore the generator can be assumed as being fairly reliable.

In Chapter 4, a normal random variable was generated using the random number sequence. To this variable, the rigorous Kolmorogov-Smirnov test was applied. This generator passed the test while some other random variable generators failed indicating the need for thorough testing before use.

In Chapter 5, programs were developed to determine the statistical properties of the excursions. Unfortunately time limitations prevented more tests from being carried out to be able to present any conclusive results.

When programs were run using the following values of β, the matrix G could not be formed.

β	Row
0.05	49
0.01	17
0.005	11

G is formed such that $GG^T = A$ where A is a N x N matrix (N = 100 in these programs). However the term SM became a small negative number on the row number indicated above. This is not permissible as the square root of SM must be determined.

Whilst it is noted that there is a trend to smaller row numbers with smaller β, this needs to be clarified.

This problem could have been due to either of the following reasons:

(i) single precision arithmetic not accurate enough

(ii) the algorithm to determine G has some form of inherent instability

Due to the error in program no. 26 and the lack of results for program no. 27 and the above, more work is required to properly relate the statistics of the excursions to those of the stochastic process.

Another important property of the excursions not evaluated in this report but which could be the subject of future work, is the integral of the excursions above the limit. This is the area between the plot of the process and the limit when the process is above the limit. The value of this area will give a measure of the average magnitude of the excursions above the limit.

CHAPTER 7

CONCLUSIONS

Using a proven random number generator, a normal random number generator was used to generate a stochastic process of given statistical properties. Using this process, it was possible to determine the statistical properties of the excursions of this process above a certain limit. Further work is required to relate the statistical properties of these excursions to those of the process.

Once this work is completed, it is expected that the results will have useful applications in engineering and other fields as given in Section 1.3 of Chapter 1.

REFERENCES

1. DOWNHAM, D.Y., AND ROBERTS, F.D.K., "Multiplicative congruential pseudo-random number generators", Computer Journal, Vol. 10, 1967, pp. 74 - 77.

2. MILLER, J.C.P., and PRENTICE, M.J., "Additive congruential pseudo-random number generators", Computer Journal, Vol. 11, 1968, pp. 341 - 346.

3. DOWNHAM, D.Y., "The runs up and down test", Computer Journal, Vol. 11, 1969, pp. 373 - 376.

4. NEVILLE, A.M., and KENNEDY, J.B., "Basic Statistical Methods for Engineers and Scientist", 1964, International Testbook Co.

5. HERRMAN, R.G., "The statistical evaluation of random number generating sequences for digital computers", Washington, D.C., Office of Technical Services, U.S. Dept. of Commerce, 1961; APEX-635.

6. GOOD, I.J., "The serial test for sampling numbers and other tests for randomness", Proc. Camb. Phil. Soc., Vol. 49, 1953, p. 276.

7. MARSAGLIA, G., "A note on the construction of a multivariate normal sample", IRE Transactions on Information Theory, June 1957, p. 149.

8. MARSAGLIA, G., "The structure of congruential sequences", Applications of number theory to numerical analysis, Academic Press, New York and London, 1972, p. 249.

9. KNUTH, D.E., "Semi-numerical algorithms", Chapter 3 of the Art of Computer Programming, Volume 2, Addison-Wesley Publishing Co.

10. MIHRAM, G.A., "Simulation - Statistical Foundations and Methodology", Academic Press, New York and London, 1972, pp. 58 - 64.

11. LINDGREN, B.W., "Statistical Theory", Macmillan, New York, 1968.

APPENDIX A

THE CHI-SQUARE (χ^2) TEST

Tests of Significance enable one to judge whether any observed differences are real or are due to chance alone. The differences studied may be those between two comparable sets of observations or between an experimental distribution and an hypothetical distribution.

1. Null Hypothesis

The Null Hypothesis postulates that there is no significant difference between the distributions being compared. The probability of the actual difference occurring due to chance alone is measured and if this probability is very small, then the null hypothesis is rejected and it is inferred that a real difference exists.

2. Significance of Test

To perform the χ^2 test, each expected class frequency E is compared with the observed frequency O, and for each class the term $\frac{(O-E)^2}{E}$ is computed. The statistic χ^2 is defined as $\chi^2 = \sum \frac{(O-E)^2}{E}$.

The calculated value is then compared with tabulated values of χ^2; the latter are values which cannot be exceeded by the calculated when there is no real difference at the specified level of significance i.e. with a given probability. Table 1 gives values of χ^2 for a range of probabilities from 0.001 to 0.99 for 1 to 30 degrees of freedom.

3. Degrees of Freedom

The number of degrees of freedom is the number of classes which can be assigned arbitarily. For example, if a coin is tossed n times, and "a" heads are observed, then number of degrees of freedom equals 1 only since number of tails is given by (n-a).

4. Too Good a Fit

Table 1 gives the values of χ^2 for probabilities of 0.001 to 0.10, that is, for levels of significance of 0.1 to 10% which are the usual levels at which the rejection of the null hypothesis is considered. The table also contains values for probabilities of 0.50 to 0.99. The occurence of χ^2 corresponding to a probability higher than about 0.99 makes one suspect that the data have been "rigged".

5. χ^2 as a Measure of Goodness of Fit

The χ^2 test is used to test the goodness of fit. In this case the null hypothesis states that there is no significant difference between the observed distribution and a postulated standard distribution.

TABLE III

Distribution of χ^2

Degrees of freedom	Probability of a deviation greater than χ^2											
	0.99	0.95	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05	0.01	0.001
1	0.000157	0.00393	0.0158	0.0642	0.148	0.455	1.074	1.642	2.706	3.841	6.635	10.827
2	0.0201	0.103	0.211	0.446	0.713	1.386	2.408	3.219	4.605	5.991	9.210	13.815
3	0.115	0.352	0.584	1.005	1.424	2.366	3.665	4.642	6.251	7.815	11.345	16.268
4	0.297	0.711	1.064	1.649	2.195	3.357	4.878	5.989	7.779	9.488	13.277	18.465
5	0.554	1.145	1.610	2.343	3.000	4.351	6.064	7.289	9.236	11.070	15.086	20.517
6	0.872	1.635	2.204	3.070	3.828	5.348	7.231	8.558	10.645	12.592	16.812	22.457
7	1.239	2.167	2.833	3.822	4.671	6.346	8.383	9.803	12.017	14.067	18.475	24.322
8	1.646	2.733	3.490	4.594	5.527	7.344	9.524	11.030	13.362	15.507	20.090	26.125
9	2.088	3.325	4.168	5.380	6.393	8.343	10.656	12.242	14.684	16.919	21.666	27.877
10	2.558	3.940	4.865	6.179	7.267	9.342	11.781	13.442	15.987	18.307	23.209	29.588
11	3.053	4.575	5.578	6.989	8.148	10.341	12.899	14.631	17.275	19.675	24.725	31.264
12	3.571	5.226	6.304	7.807	9.034	11.340	14.011	15.812	18.549	21.026	26.217	32.909
13	4.107	5.892	7.042	8.634	9.926	12.340	15.119	16.985	19.812	22.362	27.688	34.528
14	4.660	6.571	7.790	9.467	10.821	13.339	16.222	18.151	21.064	23.685	29.141	36.123
15	5.229	7.261	8.547	10.307	11.721	14.339	17.322	19.311	22.307	24.996	30.578	37.697
16	5.812	7.962	9.312	11.152	12.624	15.338	18.418	20.465	23.542	26.296	32.000	39.252
17	6.408	8.672	10.085	12.002	13.531	16.338	19.511	21.615	24.769	27.587	33.409	40.790
18	7.015	9.390	10.865	12.857	14.440	17.338	20.601	22.760	25.989	28.869	34.805	42.312
19	7.633	10.117	11.651	13.716	15.352	18.338	21.689	23.900	27.204	30.144	36.191	43.820
20	8.260	10.851	12.443	14.578	16.266	19.337	22.775	25.038	28.412	31.410	37.566	45.315
21	8.897	11.591	13.240	15.445	17.182	20.337	23.858	26.171	29.615	32.671	38.932	46.797
22	9.542	12.338	14.041	16.314	18.101	21.337	24.939	27.301	30.813	33.924	40.289	48.268
23	10.196	13.091	14.848	17.187	19.021	22.337	26.018	28.429	32.007	35.172	41.638	49.728
24	10.856	13.848	15.659	18.062	19.943	23.337	27.096	29.553	33.196	36.415	42.980	51.179
25	11.524	14.611	16.473	18.940	20.867	24.337	28.172	30.675	34.382	37.652	44.314	52.620
26	12.198	15.379	17.292	19.820	21.792	25.336	29.246	31.795	35.563	38.885	45.642	54.052
27	12.879	16.151	18.114	20.703	22.719	26.336	30.319	32.912	36.741	40.113	46.963	55.476
28	13.565	16.928	18.939	21.588	23.647	27.336	31.391	34.027	37.916	41.337	48.278	56.893
29	14.256	17.708	19.768	22.475	24.577	28.336	32.461	35.139	39.087	42.557	49.588	58.302
30	14.953	18.493	20.599	23.364	25.508	29.336	33.530	36.250	40.256	43.773	50.892	59.703

This table is abridged from Table IV of Fisher & Yates: *Statistical Tables for Biological, Agricultural, and Medical Research* published by Oliver & Boyd Ltd., Edinburgh, by permission of the authors and publishers.

To test whether an observed value of χ^2 is leading in the shaded area of the figure, i.e., exceeds the range 0 to χ^2 for a given number of degrees of freedom ν. For large ν the expression $\sqrt{2\chi^2} - \sqrt{2\nu - 1}$ may be used as a normal deviate with unit standard deviation, remembering that the probability for χ^2 corresponds to that of a single tail of the normal curve. For example, let $\chi^2 = 147.92$, $\nu = 113$; then $\sqrt{295.84} - \sqrt{225} = 5.2$, corresponding to an area = 0.4901 from Table A-4. To obtain the probability of a value exceeding χ is $0.5000 - 0.4861 = 0.0139$, which is highly significant.

APPENDIX B

COMPUTER PROGRAMS NOS. 1 TO 27

The following subroutines are used in more than one program. To avoid repetition they are shown in only the first program in which they occur.

Subroutine	Program No.
Function Uni (J)	1
Subroutine K S test	9
Subroutine Asort	9
Function ERF (Y)	12
Function RNOR (JJ)	14
Subroutine PLOT	17
Subroutine CROSS	19
Subroutine DECOMP	21

PROGRAM No. 1

Name: RAND

Purpose: Program for generation of a random number sequence.

PROGRAM RAND CDC 6600 FTN V3.0

```
      PROGRAM RAND (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
      DIMENSION RAN(128),X(100)
      COMMON/MNP/FCT
      FCT=2.**(-48)
    5 ST=UNI(1234567)
      DO 1 I=1,100
      X(I)=UNI(0)
    1 WRITE(61,3)X(I)
    3 FORMAT(3X,E13.6)
   10 STOP
      END
```

FUNCTION UNI CDC 6600 FTN V3.0

```
      FUNCTION UNI(JJ)
      DIMENSION RAN(128)
      COMMON/MNP/FCT
      DATA I,K/6832853,123/
    5 IF(JJ.NE.0)GOTO4
      K=MOD(K*69,128)+1
      UNI=RAN(K)
      I=I*69
      RAN(K)=I*FCT
   10 RETURN
    4 DO2L=1,128
      I=I*69
    2 RAN(L)=I*FCT
      UNI=0.
   15 RETURN
      END
```

FUNCTION UNI CDC 6600 FTN

SYMBOLIC REFERENCE MAP

ENTRY POINTS
2 UNI

VARIABLES	SN	TYPE	RELOCATION			
0	FCT	REAL	MNP	40	I	INTE
0	JJ	INTEGER	F.P.	41	K	INTE
43	L	INTEGER		44	RAN	REAL
42	UNI	REAL				

INLINE FUNCTIONS	TYPE	ARGS	
MOD	INTEGER	2	INTRIN

STATEMENT LABELS
0 2 26 4

COMMON BLOCKS	LENGTH
MNP	1

STATISTICS
PROGRAM LENGTH 244B 164
COMMON LENGTH 1B 1

.967433E+00
.924702E+00
.271341E+00
.663232E+00
.458762E+00
.459324E+00
.118278E+00
.976563E+00
.361386E+00
.597377E+00
.627093E+00
.961518E-01
.301037E+00
.553812E+00
.198028E+00
.122833E+00
.645141E+00
.439997E+00
.614910E+00
.132674E+00
.836546E-01
.885927E+00
.323411E-01
.662721E+00
.438235E+00
.329598E+00
.800210E-01
.217912E+00
.945772E+00
.987594E-01
.824175E+00
.770237E+00
.934070E-01
.626273E+00
.580532E+00
.915139E+00
.171870E+00
.936285E+00
.247035E+00

.106316E+00
.167666E-02
.122065E+00
.372224E+00
.580523E+00
.879810E+00
.805350E+00

.781208E+00
.193204E-01
.604642E+00
.626923E-01
.405928E+00
.240831E+00

.192507E-01
.127154E+00
.580437E+00
.564648E+00
.284316E+00
.286347E+00

.787176E+00
.287378E+00
.293235E+00
.563284E+00
.387284E+00
.761245E+00

.524792E+00
.652340E+00
.355166E+00
.749119E+00
.982416E+00
.495320E-01

.765005E+00
.156281E+00
.205586E+00
.611206E+00
.503116E+00
.194952E+00

.524543E+00
.644436E+00
.343461E+00
.607662E+00
.598765E+00
.214168E+00

.973383E+00
.806200E+00
.247826E+00
.857406E+00
.693694E+00
.149247E+00

.436952E+00
.492321E+00
.446388E+00
.506675E+00
.135222E+00
.377559E-01

.142996E+00
.703817E+00
.405582E+00
.330692E+00
.627287E+00

PROGRAM No. 2

Name: UNIG

Purpose: Program for testing the uniformity of a random number sequence.

```
PROGRAM   UNIG                                          CDC 6600 FTN V3.0-F

      PROGRAM UNIG (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPUT)
      DIMENSIONX(1000),ICOUNT(10)
      COMMON/MNP/FCT
      FCT=2.**(-48)
 5    DATA CHI,EXP/0.0,100.0/
      DO20J=1,10
   20 ICOUNT(J)=0
      ST=UNI(1234567)
      DO10I=1,1000
 10   X(I)=UNI(0)
      IC=10.*X(I)+1
   10 ICOUNT(IC)=ICOUNT(IC)+1
      DO19K=1,10
   19 WRITE(61,18)ICOUNT(K)
 15   18 FORMAT(I6)
      DO21J=1,10
   21 CHI=CHI+(ICOUNT(J)-EXP)**2/EXP
      WRITE(61,22)CHI
   22 FORMAT(3X,E16.7)
 20   STOP
      END
```

```
 86
119
 93
105
 89
100
111
108
 90
 99
     .1038000E+02
```

PROGRAM No. 3

Name: AMBD

Purpose: Program for testing the number of runs above and below the median of a random number sequence.

PROGRAM ABMD CDC 6600 FTN V3.1

```
      PROGRAM ABMD (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
      DIMENSIONX(1000),R(10),RE(10),EL(10)
      COMMON/MNP/FCT
      FCT=2.**(-48)
    5 DO50I=1,10
      R(I)=0.0
   50 EL(I)=0.0
      READ(60,51)RE
   51 FORMAT(10F8.0)
   10 READ(60,52)CHI
   52 FORMAT(F6.0)
      ST=UNI(1234567)
      DO10I=1,1000
   10 X(I)=UNI(0)
   15 I=1
   11 IF(I.GE.1000)GOTO31
      IF(X(I).LT.0.5)GOTO12
      IA=I+1
      IF(X(IA).LT.0.5)GOTO13
   20 IF(IA.GE.1000)GOTO13
      IB=IA+1
      IF(X(IB).LT.0.5)GOTO14
      IF(IB.GE.1000)GOTO14
      IC=IB+1
   25 IF(X(IC).LT.0.5)GOTO15
      IF(IC.GE.1000)GOTO15
      ID=IC+1
      IF(X(ID).LT.0.5)GOTO16
      IF(ID.GE.1000)GOTO16
   30 IE=ID+1
      IF(X(IE).LT.0.5)GOTO17
      IF(IE.GE.1000)GOTO17
      IP=IE+1
      IF(X(IR).LT.0.5)GOTO18
   35 IF(IR.GE.1000)GOTO18
      IS=IR+1
      IF(X(IS).LT.0.5)GOTO19
      IF(IS.GE.1000)GOTO19
      IT=IS+1
   40 IF(X(IT).LT.0.5)GOTO20
      IF(IT.GE.1000)GOTO20
      IU=IT+1
      IF(X(IU).LT.0.5)GOTO21
   22 R(10)=R(10)+1.0
   45 I=IU+1
      GOTO11
   12 IA=I+1
      IF(X(IA).GE.0.5)GOTO13
      IF(IA.GE.1000)GOTO13
   50 IB=IA+1
      IF(X(IB).GE.0.5)GOTO14
      IF(IB.GE.1000)GOTO14
      IC=IB+1
      IF(X(IC).GE.0.5)GOTO15
   55 IF(IC.GE.1000)GOTO15
```

PROGRAM AHMD CDC 6600 FTN V3.

```
      ID=IC+1
      IF(X(ID).GE.0.5)GOTO16
      IF(ID.GE.1000)GOTO16
      IF=ID+1
60    IF(X(IF).GE.0.5)GOTO17
      IF(IF.GE.1000)GOTO17
      IR=IF+1
      IF(X(IR).GE.0.5)GOTO18
      IF(IR.GE.1000)GOTO18
65    IS=IR+1
      IF(X(IS).GE.0.5)GOTO19
      IF(IS.GE.1000)GOTO19
      IT=IS+1
      IF(X(IT).GE.0.5)GOTO20
70    IF(IT.GE.1000)GOTO20
      IU=IT+1
      IF(X(IU).GE.0.5)GOTO21
      GOTO22
   13 R(1)=R(1)+1.0
75    I=IA
      GOTO11
   14 R(2)=R(2)+1.0
      I=IB
      GOTO11
80 15 R(3)=R(3)+1.0
      I=IC
      GOTO11
   16 R(4)=R(4)+1.0
      I=ID
85    GOTO11
   17 R(5)=R(5)+1.0
      I=IE
      GOTO11
   18 R(6)=R(6)+1.0
90    I=IR
      GOTO11
   19 R(7)=R(7)+1.0
      I=IS
      GOTO11
95 20 R(8)=R(8)+1.0
      I=IT
      GOTO11
   21 R(9)=R(9)+1.0
      I=IU
100   GOTO11
   31 DO32I=1,10
      EL(I)=(R(I)-RE(I))**2/RE(I)
   32 CHI=CHI+EL(I)
      WRITE(6,38)CHI
105 38 FORMAT(3X,E16.7)
      WRITE(6,37)(R(I),I=1,10)
   37 FORMAT(3X,E16.7)
      STOP
      END
```

FUNCTION UNI CDC 6600 FTN V3

SYMBOLIC REFERENCE MAP

ENTRY POINTS
2 UNI

VARIABLES	SN	TYPE	RELOCATION			
0	FCT	REAL	MNP	40	I	INTEGE
0	JJ	INTEGER	F.P.	41	K	INTEGE
43	L	INTEGER		44	RAN	REAL
42	UNI	REAL				

INLINE FUNCTIONS	TYPE	ARGS	
MOD	INTEGER	2	INTRIN

STATEMENT LABELS
0 2 26 4

COMMON BLOCKS	LENGTH
MNP	1

STATISTICS

PROGRAM LENGTH	244B	164
COMMON LENGTH	1B	1

.5035875E+01
.2490000E+03
.1270000E+03
.6800000E+02
.2800000E+02
.1200000E+02
.1000000E+02
.1000000E+01
.3000000E+01
.1000000E+01
.2000000E+01

PROGRAM No. 4

Name: UPDO

Purpose: Program for testing the number of runs up and down of a random number sequence.

```
PROGRAM    UPDO                                          CDC 6600 FTN V3.(

           PROGRAM UPDO (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
           DIMENSIONX(1000),RU(5),RD(5),R(5),RE(5),EL(5)
           COMMON/MNP/FCT
           FCT=2.**(-48)
 5         DO33I=1,5
           RU(I)=0.0
           RD(I)=0.0
           R(I)=0.0
        33 EL(I)=0.0
10         READ(60,34)RE
        34 FORMAT(5F10.0)
           READ(60,35)CHI
        35 FORMAT(F6.0)
           ST=UNI(1234567)
15         DO10I=1,1000
        10 X(I)=UNI(0)
           I=1
        11 IF(I.GE.1000)GOTO31
           J=I+1
20         IF(X(J)-X(I))12,13,14
        14 IF(J.GE.1000)GOTO15
           K=J+1
           IF(X(K)-X(J))15,13,16
        15 RU(1)=RU(1)+1.0
25         I=J
           GOTO11
        16 IF(K.GE.1000)GOTO17
           L=K+1
           IF(X(L)-X(K))17,13,18
30      17 RU(2)=RU(2)+1.0
           I=K
           GOTO11
        18 IF(L.GE.1000)GOTO19
           M=L+1
35         IF(X(M)-X(L))19,13,20
        19 RU(3)=RU(3)+1.0
           I=L
           GOTO11
        20 IF(M.GE.1000)GOTO21
40         N=M+1
           IF(X(N)-X(M))21,13,22
        21 RU(4)=RU(4)+1.0
           I=M
           GOTO11
45      22 RU(5)=RU(5)+1.0
           I=N
           GOTO11
        12 IF(J.GE.1000)GOTO24
           K=J+1
50         IF(X(K)-X(J))23,13,24
        24 RD(1)=RD(1)+1.0
           I=J
           GOTO11
        23 IF(K.GE.1000)GOTO26
55         L=K+1
```

PROGRAM UPDO CDC 6600 FTN V3.0

```
      IF(X(L)-X(K))25,13,26
   26 RD(2)=RD(2)+1.0
      I=K
      GOTO11
   25 IF(L.GE.1000)GOTO28
      M=L+1
      IF(X(M)-X(L))27,13,28
   28 RD(3)=RD(3)+1.0
      I=L
      GOTO11
   27 IF(M.GE.1000)GOTO30
      N=M+1
      IF(X(N)-X(M))29,13,30
   30 RD(4)=RD(4)+1.0
      I=M
      GOTO11
   29 RD(5)=RD(5)+1.0
      I=N
      GOTO11
   13 WRITE(61,36)(I,X(I),X(J),X(K),X(L),X(M),X(N))
   36 FORMAT(I6,6(2X,E16.9))
   31 DO32I=1,5
      R(I)=RU(I)+RD(I)
      EL(I)=(R(I)-RE(I))**2/RE(I)
   32 CHI=CHI+EL(I)
      WRITE(61,37)(R(I),RU(I),RD(I),I=1,5)
   37 FORMAT(3(3X,E16.7))
      WRITE(61,38)CHI
   38 FORMAT(3X,E16.7)
      STOP
      END
```

```
 .4060000E+03      .2090000E+03      .1970000E+03
 .1960000E+03      .9400000E+02      .1020000E+03
 .5700000E+02      .2400000E+02      .3300000E+02
 .5000000E+01      .4000000E+01      .1000000E+01
 .2000000E+01      .2000000E+01      0.
 .4461042E+01
```

PROGRAM No. 5

Name: SERIAL

Purpose: Program for testing the serial lag properties of a random number sequence.

PROGRAM SERIAL CDC 6600 FTN V3.0

```
      PROGRAM SERIAL (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUT
      DIMENSION X(1006),G(10),F(10,10),A(10),B(10)
      COMMON/MNP/FCT
      FCT=2.**(-48)
    5 ST=UNI(1234567)
      DO10I=1,1006
   10 X(I)=UNI(0)
      DO11L=1,6
      DO50M=1,10
   10 G(M)=0.0
      DO50N=1,10
   50 F(M,N)=0.0
      DO21IA=1,1000
      K=IA+L
   15 J=1
      A(1)=0.1
   16 IF(X(IA).GE.A(J))GOTO14
      M=J
      G(M)=G(M)+1.0
   20 GOTO13
   14 J=J+1
      IF(J.GT.10)GOTO24
      JJ=J-1
      A(J)=A(JJ)+0.1
   25 GOTO16
   13 JA=1
      B(1)=0.1
   17 IF(X(K).GE.B(JA))GOTO18
      N=JA
   30 GOTO20
   18 JA=JA+1
      IF(JA.GT.10)GOTO24
      JB=JA-1
      B(JA)=B(JB)+0.1
   35 GOTO17
   20 F(M,N)=F(M,N)+1.0
   21 CONTINUE
      DO22M=1,10
   22 WRITE(61,23)(F(M,N),N=1,10)
   40 23 FORMAT(10(2X,F5.0))
      Z=0.0
      Y=0.0
      DO30M=1,10
      Z=Z+(G(M)-200)**2
   45 DO30N=1,10
   30 Y=Y+(F(M,N)-20)**2
      CHI=Y/20-Z/200
      WRITE(61,38)CHI
   38 FORMAT(3X,E16.7)
   50 11 CONTINUE
   24 STOP
      END
```

3.	14.	7.	10.	9.	6.	17.	7.	10.	3.
11.	13.	10.	11.	11.	19.	12.	11.	10.	11.
7.	10.	9.	9.	8.	7.	8.	9.	12.	14.
5.	15.	13.	14.	9.	14.	10.	10.	5.	10.
6.	10.	7.	10.	11.	7.	8.	11.	12.	7.
9.	14.	8.	13.	7.	6.	13.	11.	9.	10.
9.	14.	8.	14.	12.	14.	16.	9.	5.	10.
13.	10.	16.	9.	11.	8.	8.	11.	9.	13.
10.	10.	7.	5.	5.	9.	11.	14.	11.	8.
13.	9.	8.	10.	6.	10.	8.	15.	8.	12.
.4061000E+02									
10.	12.	10.	10.	4.	10.	8.	10.	5.	7.
7.	18.	8.	14.	8.	14.	19.	8.	9.	14.
9.	8.	6.	9.	9.	11.	13.	6.	11.	11.
9.	12.	12.	14.	7.	10.	10.	15.	8.	8.
9.	9.	11.	5.	10.	10.	16.	9.	7.	3.
4.	15.	12.	8.	11.	8.	7.	13.	10.	12.
11.	13.	5.	18.	12.	8.	10.	12.	14.	8.
10.	4.	10.	11.	12.	12.	11.	11.	13.	14.
10.	13.	9.	6.	6.	7.	4.	17.	8.	10.
7.	15.	10.	10.	10.	10.	13.	7.	6.	11.
.4691000E+02									
5.	11.	7.	12.	10.	7.	11.	8.	7.	8.
11.	12.	11.	10.	10.	21.	14.	16.	10.	4.
7.	10.	9.	9.	10.	9.	7.	10.	10.	12.
10.	10.	10.	9.	12.	9.	13.	15.	9.	8.
8.	15.	9.	8.	7.	6.	12.	7.	4.	13.
11.	9.	14.	9.	8.	7.	16.	9.	8.	9.
5.	15.	10.	15.	8.	16.	9.	14.	9.	10.
12.	13.	7.	11.	6.	11.	15.	6.	18.	9.
11.	8.	11.	11.	10.	8.	6.	10.	3.	12.
6.	16.	4.	11.	8.	6.	8.	13.	14.	13.
.4721000E+02									
9.	9.	9.	9.	6.	6.	15.	9.	7.	7.
7.	17.	11.	12.	10.	9.	14.	9.	16.	14.
7.	12.	12.	10.	12.	8.	4.	9.	9.	10.
16.	9.	9.	11.	10.	10.	11.	13.	7.	
3.	11.	8.	8.	11.	11.	11.	11.	9.	
5.	11.	13.	5.	7.	14.	12.	16.	6.	
13.	15.	9.	15.	6.	16.	9.	7.	12.	9.
5.	10.	10.	11.	10.	12.	13.	12.	12.	13.
10.	10.	3.	10.	11.	6.	9.	15.	8.	8.
11.	9.	10.	14.	11.	8.	10.	12.	6.	8.
.4321000E+02									
6.	7.	10.	8.	11.	9.	7.	11.	9.	8.
13.	13.	9.	15.	10.	12.	11.	16.	13.	7.
8.	14.	8.	7.	6.	6.	12.	14.	7.	11.
9.	10.	9.	15.	12.	11.	12.	5.	15.	7.
9.	6.	6.	4.	6.	10.	10.	13.	11.	14.
10.	13.	9.	10.	11.	11.	11.	9.	7.	9.
7.	12.	16.	19.	8.	12.	8.	13.	6.	10.
6.	13.	11.	14.	10.	6.	12.	10.	10.	16.
11.	12.	6.	6.	9.	11.	9.	10.	8.	8.
7.	19.	9.	7.	5.	12.	18.	7.	6.	9.
.4511000E+02									
15.	6.	9.	10.	5.	9.	7.	8.	7.	10.
9.	16.	11.	14.	12.	6.	15.	10.	15.	11.
12.	10.	9.	10.	8.	11.	14.	8.	4.	7.
4.	16.	17.	10.	8.	15.	9.	10.	5.	11.
7.	13.	7.	7.	6.	10.	9.	8.	14.	8.
7.	11.	10.	9.	9.	10.	11.	13.	9.	11.
9.	12.	8.	13.	8.	11.	14.	10.	16.	10.
6.	13.	6.	11.	14.	14.	8.	15.	10.	11.
8.	12.	12.	10.	8.	6.	9.	13.	7.	5.
9.	10.	4.	12.	9.	8.	14.	13.	5.	15.

PROGRAM No. 6

Name: SUMN

Purpose: Program for testing the sum of N terms properties of a random number sequence.

PROGRAM SUMN CDC 6600 FTN V3.0-P296 OPT=1

```
      PROGRAM SUMN (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPUT)
      DIMENSION X(1000),S(500),T(333),U(250),V(200),SUM(10),A(10)
      COMMON/MNP/FCT
      FCT=2.**(-48)
      ST=UNI(1234567)
      DO10I=1,1000
   10 X(I)=UNI(0)
      DO11K=1,500
      K1=2*K-1
      K2=2*K
   11 S(K)=(X(K1)+X(K2))/2.
      DO20J=1,10
   20 SUM(J)=0.0
      DO13I=1,500
      J=1
      A(1)=0.1
   16 IF(S(I).GE.A(J))GOTO14
      SUM(J)=SUM(J)+1.0
      GOTO13
   14 J=J+1
      IF(J.GT.10)STOP
      K=J-1
      A(J)=A(K)+0.1
      GOTO16
   13 CONTINUE
      DO17J=1,10
   17 WRITE(61,18)SUM(J)
   18 FORMAT(3X,E16.7)
      CHI=(SUM(1)-10.)**2/10.+(SUM(2)-30.)**2/30.+(SUM(3)-50.)**2/50.
     0+(SUM(4)-70.)**2/70.+(SUM(5)-90.)**2/90.+(SUM(6)-90.)**2/90.
     0+(SUM(7)-70.)**2/70.+(SUM(8)-50.)**2/50.+(SUM(9)-30.)**2/30.
     0+(SUM(10)-10.)**2/10.
      WRITE(61,22)CHI
   22 FORMAT(3X,E16.7)
      DO31K=1,333
      K1=3*K-2
      K2=3*K-1
      K3=3*K
   31 T(K)=(X(K1)+X(K2)+X(K3))/3.
      DO40J=1,10
   40 SUM(J)=0.0
      DO33I=1,333
      J=1
      A(1)=0.1
   36 IF(T(I).GE.A(J))GOTO34
      SUM(J)=SUM(J)+1.0
      GOTO33
   34 J=J+1
      IF(J.GT.10)STOP
      K=J-1
      A(J)=A(K)+0.1
      GOTO36
   33 CONTINUE
      DO37J=1,10
   37 WRITE(61,38)SUM(J)
```

```
PROGRAM      SUMN                                    CDC 6600 FTN V3.0-P296 OPT=1   7
      38 FORMAT(3X,E16.7)
         CHI=(SUM(1)-1.5)**2/1.5+(SUM(2)-10.5)**2/10.5
        Q+(SUM(3)-28.5)**2/28.5+(SUM(4)-54.1)**2/54.1+(SUM(5)-71.9)**2/71.9
        Q+(SUM(6)-71.9)**2/71.9+(SUM(7)-54.1)**2/54.1+(SUM(8)-28.5)**2/28.5
      Q+(SUM(9)-10.5)**2/10.5+(SUM(10)-1.5)**2/1.5
      WRITE(61,42)CHI
      42 FORMAT(3X,E16.7)
         DO51K=1,250
         K1=4*K-3
         K2=4*K-2
         K3=4*K-1
         K4=4*K
      51 U(K)=(X(K1)+X(K2)+X(K3)+X(K4))/4.
         DO60J=1,10
      60 SUM(J)=0.0
         DO53I=1,250
         J=1
         A(1)=0.1
      56 IF(U(I).GE.A(J))GOTO54
         SUM(J)=SUM(J)+1.0
         GOTO53
      54 J=J+1
         IF(J.GT.10)STOP
         K=J-1
         A(J)=A(K)+0.1
         GOTO56
      53 CONTINUE
         DO57J=1,10
      57 WRITE(61,58)SUM(J)
      58 FORMAT(3X,E16.7)
         CHI=(SUM(1)-0.27)**2/0.27+(SUM(2)-4.0)**2/4.0
        Q+(SUM(3)-17.2)**2/17.2+(SUM(4)-41.4)**2/41.4+(SUM(5)-62.1)**2/62.1
        Q+(SUM(6)-62.1)**2/62.1+(SUM(7)-41.4)**2/41.4+(SUM(8)-17.2)**2/17.2
        Q+(SUM(9)-4.0)**2/4.0+(SUM(10)-0.27)**2/0.27
         WRITE(61,62)CHI
      62 FORMAT(3X,E16.7)
         DO71K=1,200
         K1=5*K-4
         K2=5*K-3
         K3=5*K-2
         K4=5*K-1
         K5=5*K
      71 V(K)=(X(K1)+X(K2)+X(K3)+X(K4)+X(K5))/5.
         DO80J=1,10
      80 SUM(J)=0.0
         DO73I=1,200
         J=1
         A(1)=0.1
      76 IF(V(I).GE.A(J))GOTO74
         SUM(J)=SUM(J)+1.0
         GOTO73
      74 J=J+1
         IF(J.GT.10)STOP
         K=J-1
         A(J)=A(K)+0.1
```

```
PROGRAM      SUMN                                    CDC 6600 FTN V3.0-P296 OPT=1
                GO1076
             73 CONTINUE
                DO77J=1,10
             77 WRITE(61,78)SUM(J)
15           78 FORMAT(3X,E16.7)
                CHI=(SUM(1)-0.05)**2/0.05+(SUM(2)-1.62)**2/1.62
               0+(SUM(3)-10.7)**2/10.7+(SUM(4)-32.6)**2/32.6+(SUM(5)-55.0)**2/55.0
               0+(SUM(6)-55.0)**2/55.0+(SUM(7)-32.6)**2/32.6+(SUM(8)-10.7)**2/10.7
               0+(SUM(9)-1.62)**2/1.62+(SUM(10)-0.05)**2/0.05
20              WRITE(61,82)CHI
             82 FORMAT(3X,F16.7)
                STOP
                END
```

VARIABLES	SN	TYPE	RELOCATION			
0	FCT	REAL	MNP	40	I	INTEGE
0	JJ	INTEGER	F.P.	41	K	INTEGE
43	L	INTEGER		44	RAN	REAL
42	UNI	REAL				

INLINE FUNCTIONS	TYPE	ARGS	
MOD	INTEGER	2	INTRIN

STATEMENT LABELS

0 2 26 4

COMMON BLOCKS	LENGTH
MNP	1

STATISTICS

PROGRAM LENGTH	244B	164
COMMON LENGTH	1B	1

.6000000E+01
.3400000E+02
.4100000E+02
.8200000E+02
.8700000E+02

.8400000E+02
.6500000E+02
.5700000E+02
.3300000E+02
.1100000E+02
.8047619E+01

0.
.9000000E+01
.2600000E+02
.4900000E+02
.8500000E+02
.7900000E+02

.4700000E+02
.2300000E+02
.1400000E+02
.1000000E+01
.8828790E+01

0.
.2000000E+01
.1700000E+02
.4600000E+02
.5800000E+02
.5700000E+02
.4600000E+02

.1900000E+02
.5000000E+01

0.
.3692453E+01

0.
.1000000E+01

.9000000E+01
.3800000E+02
.4900000E+02
.5500000E+02
.3800000E+02
.8000000E+01

.2000000E+01

0.
.3821324E+01

PROGRAM No. 7

Name: POKF

Purpose: Program for the poker test for a random number sequence.

PROGRAM POKF CDC 6600 FTN V3.0-P

```
      PROGRAM POKF (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPUT)
      DIMENSION X(1000),IX(1000)
      COMMON/MNP/FCT
      FCT=2.**(-48)
    5 ST=UNI(1234567)
      DO10I=1,1000
      X(I)=UNI(0)
   10 IX(I)=10.*X(I)+1
      BUST=0.0
   10 PAIR=0.0
      TWOP=0.0
      TRIO=0.0
      FULL=0.0
      FOUR=0.0
   15 DO99K=1,996,5
      K1=K+1
      K2=K+2
      K3=K+3
      K4=K+4
   20 A=IX(K)
      B=IX(K1)
      C=IX(K2)
      D=IX(K3)
      E=IX(K4)
   25 IF(A.EQ.B)GOTO30
      IF(A.EQ.C)GOTO31
      IF(A.EQ.D)GOTO32
      IF(A.EQ.E)GOTO33
      IF(B.EQ.C)GOTO34
   30 IF(B.EQ.D)GOTO35
      IF(B.EQ.E)GOTO36
      IF(C.EQ.D)GOTO37
      IF(C.EQ.E)GOTO94
      IF(D.EQ.E)GOTO94
   35 BUST=BUST+1.
      GOTO99
   30 IF(A.EQ.C)GOTO40
      IF(A.EQ.D)GOTO41
      IF(A.EQ.E)GOTO42
   40 IF(C.EQ.D)GOTO43
      IF(C.EQ.E)GOTO95
      IF(D.EQ.E)GOTO95
      GOTO94
   40 IF(A.EQ.D)GOTO98
   45 IF(A.EQ.E)GOTO98
      IF(D.EQ.E)GOTO97
      GOTO96
   41 IF(A.EQ.E)GOTO98
      IF(C.EQ.E)GOTO97
   50 GOTO96
   42 IF(C.EQ.D)GOTO97
      GOTO96
   43 IF(D.EQ.E)GOTO97
      GOTO95
   55 31 IF(A.EQ.D)GOTO44
```

```
      PROGRAM    POKF                                        CDC 6600 FTN V3.0-P
                       IF(A.EQ.E)GOTO45
                       IF(B.EQ.D)GOTO46
                       IF(B.EQ.E)GOTO95
                       IF(D.EQ.E)GOTO95
           60          GOTO94
                   44  IF(A.EQ.E)GOTO96
                       IF(B.EQ.E)GOTO97
                       GOTO96
                   45  IF(B.EQ.D)GOTO97
           65          GOTO96
                   46  IF(D.EQ.E)GOTO97
                       GOTO95
                   32  IF(A.EQ.E)GOTO47
                       IF(B.EQ.C)GOTO48
           70          IF(B.EQ.E)GOTO95
                       IF(C.EQ.E)GOTO95
                       GOTO94
                   47  IF(B.EQ.C)GOTO97
                       GOTO96
           75      48  IF(C.EQ.E)GOTO97
                       GOTO95
                   33  IF(B.EQ.C)GOTO49
                       IF(B.EQ.D)GOTO95
                       IF(C.EQ.D)GOTO95
           80          GOTO94
                   49  IF(B.EQ.D)GOTO97
                       GOTO95
                   34  IF(B.EQ.D)GOTO50
                       IF(B.EQ.E)GOTO96
           85          IF(D.EQ.E)GOTO95
                       GOTO94
                   50  IF(B.EQ.E)GOTO98
                       GOTO96
                   35  IF(B.EQ.E)GOTO96
           90          IF(C.EQ.E)GOTO95
                       GOTO94
                   36  IF(C.EQ.D)GOTO95
                       GOTO94
                   37  IF(C.EQ.E)GOTO96
           95          GOTO94
                   94  PAIR=PAIR+1.
                       GOTO99
                   95  TWOP=TWOP+1.
                       GOTO99
          100      96  TRIO=TRIO+1.
                       GOTO99
                   97  FULL=FULL+1.
                       GOTO99
                   98  FOUR=FOUR+1.
          105      99  CONTINUE
                       WRITE(61,78)BUST,PAIR,TWOP,TRIO,FULL,FOUR
                   78  FORMAT(3X,E12.3)
                       CHI=((BUST-69.48)**2)/69.48+((PAIR-100.8)**2)/100.8
                      0+((TWOP-21.6)**2)/21.6+((TRIO-14.4)**2)/14.4
          110         0+((FULL-1.08-1)/(1.0*((FOUR-0.92)**2)/0.92
```

PROGRAM POKF CDC 6600 FTN V3.0-P

```
      WRITE(61,71)CHI
   71 FORMAT(3X,E16.7)
      STOP
      END
```

```
     ,660E+02
     ,900E+02
     ,200E+02
     ,150E+02
     ,300E+01
    0,
     ,2595899E+01
```

PROGRAM No. 8

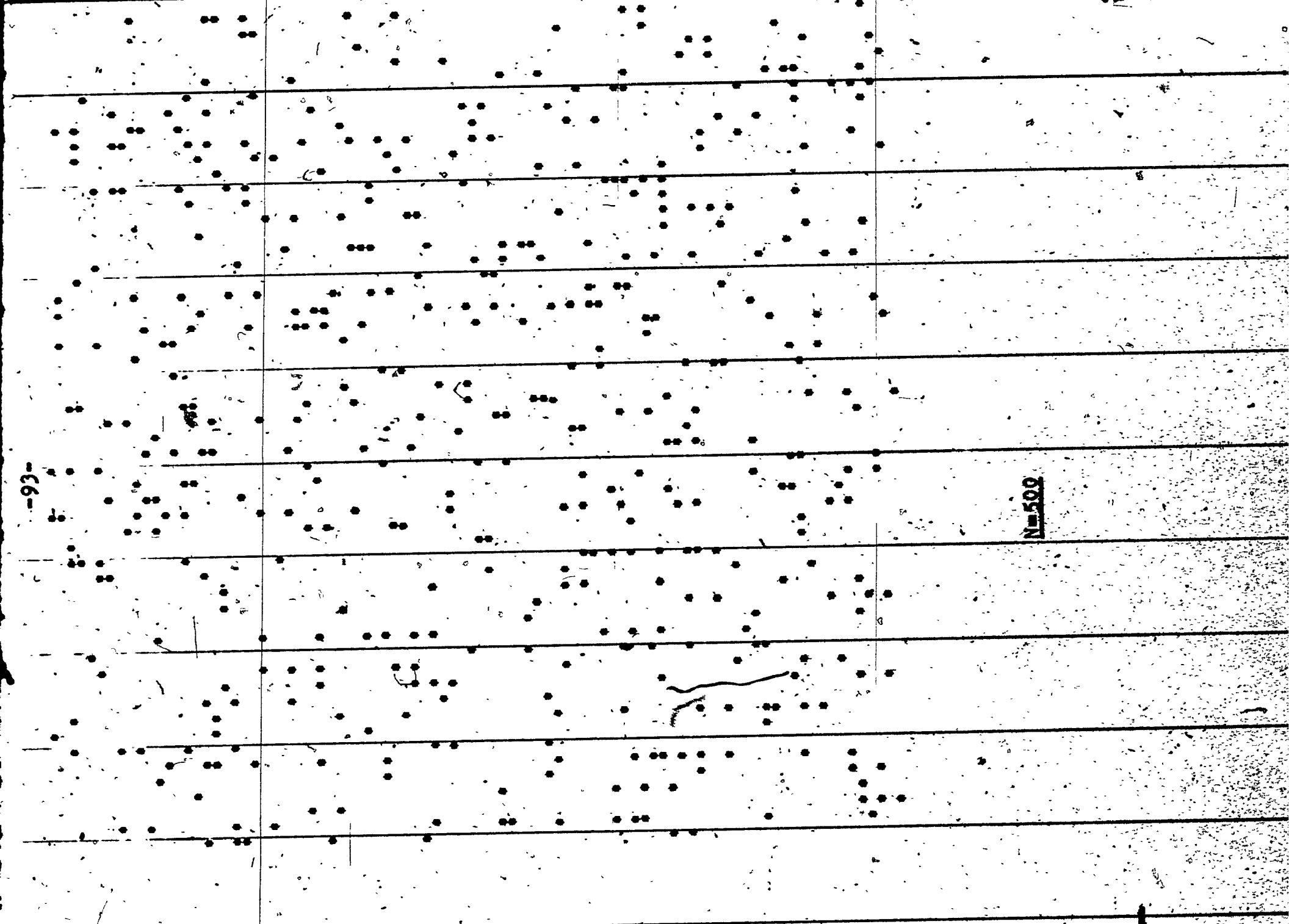
Name: PLOT1

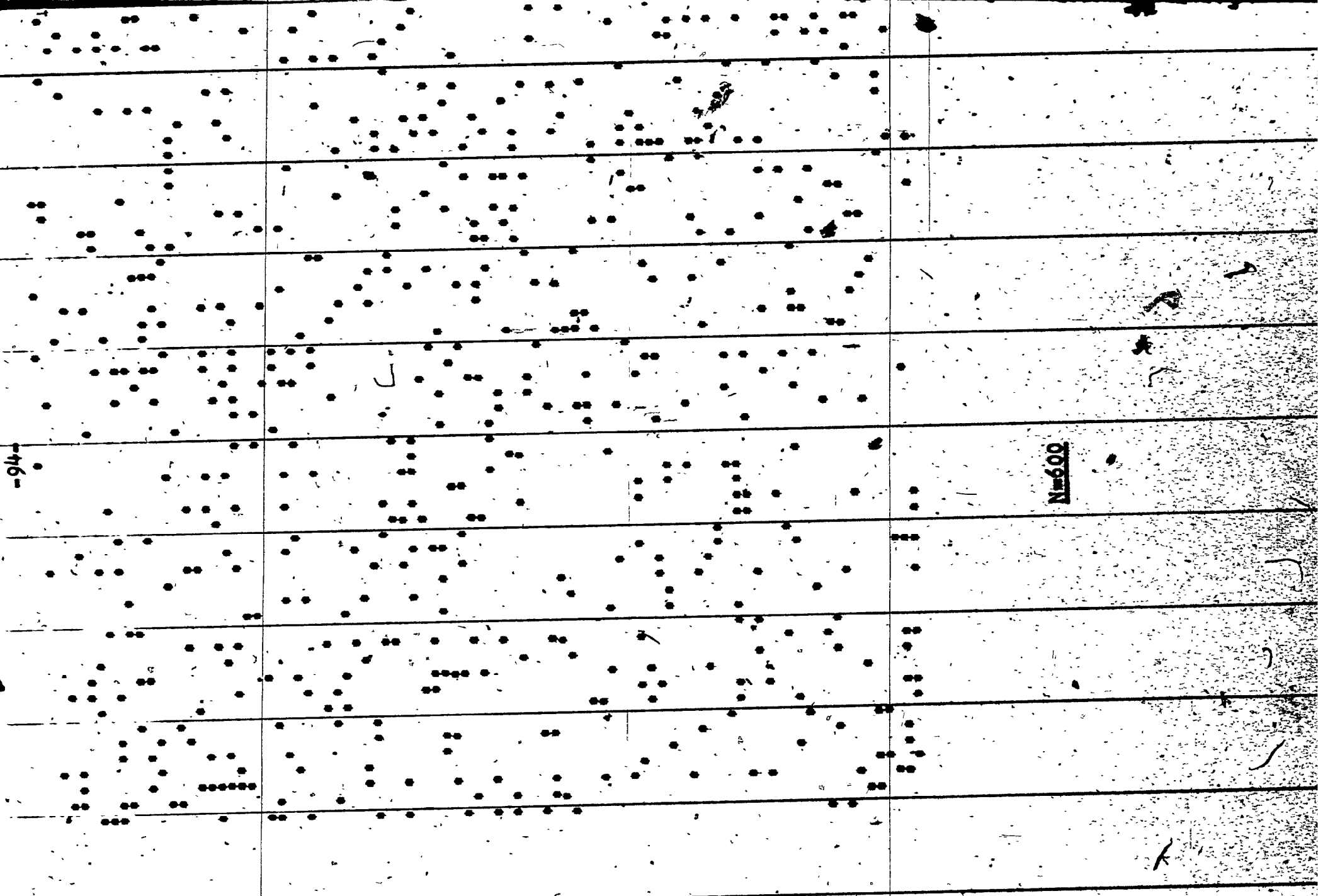
Purpose: Program for plotting points with coordinates which are consecutive numbers in a random number sequence.

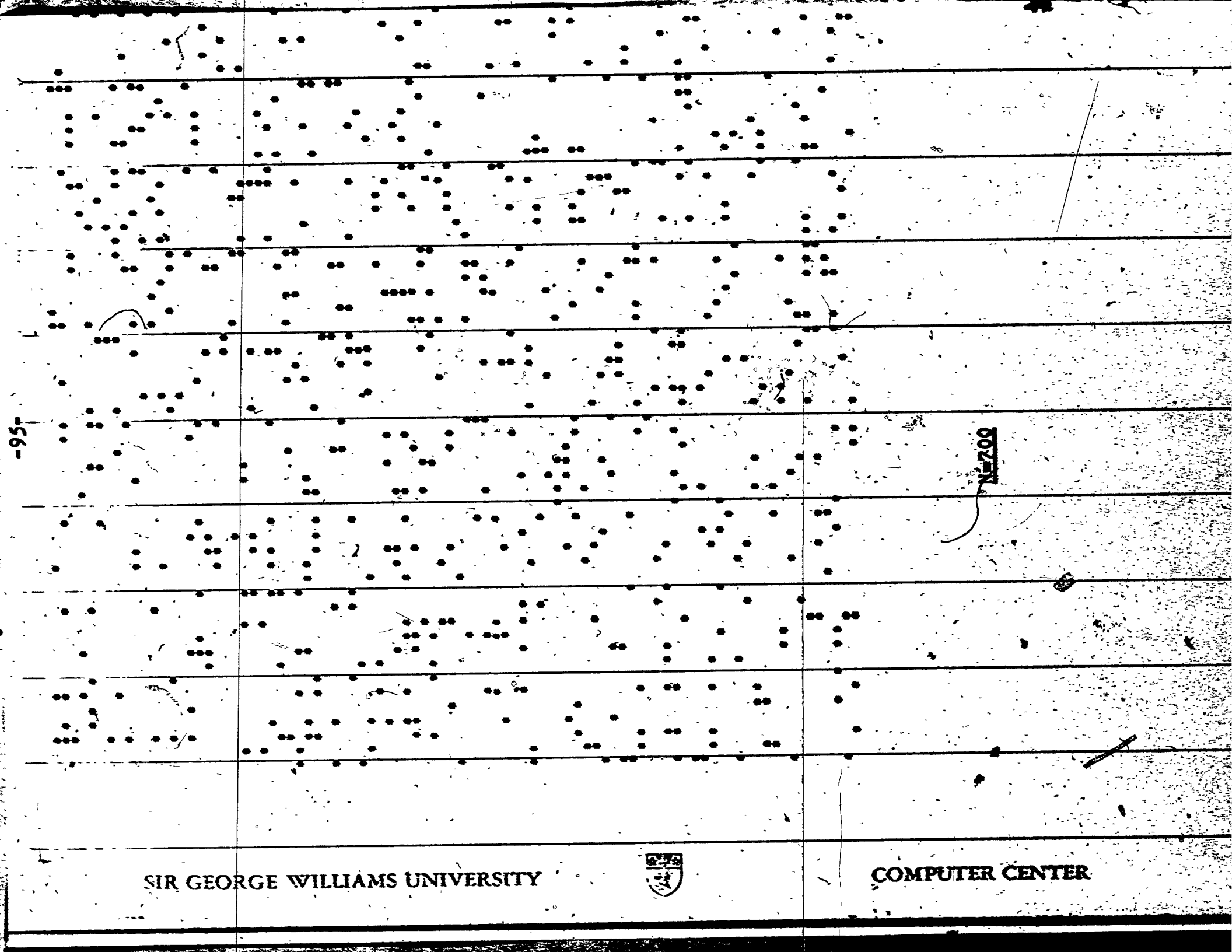
PROGRAM PLOT1 CDC 6600 FTN V3.0-

```
      PROGRAM PLOT1 (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
      DIMENSIONX(1000),Y(1000)
      INTEGER BLANK,STAR
      COMMON/MNP/FCT
    5 COMMON/MNQ/BLANK,STAR
      DATA BLANK,STAR/1H ,1H*/
      FCT=2.**(-48)
      DO99N=500,1000,100
      ST=UNI(1234567)
   10 DO 10 I=1,N
      X(I)=UNI(0)
   10 Y(I)=UNI(0)
      CALL PLOT(X,Y,N)
   99 CONTINUE
   15 STOP
      END
```

```
      SUBROUTINE PLOT(X,Y,N)
      DIMENSION X(1000),Y(1000),M(90,54)
      INTEGER BLANK,STAR
    5 COMMON/MNQ/BLANK,STAR
      DO11I=1,90
      DO11J=1,54
   11 M(I,J)=BLANK
      DO12I=1,N
   10 IX=90.*X(I)+1
      IY=54.*Y(I)+1
   12 M(IX,IY)=STAR
      WRITE(61,30)
   30 FORMAT(1H1)
   15 DO13I=1,54
   13 WRITE(61,14)(M(J,I),J=1,90)
   14 FORMAT(3X,90A1)
      RETURN
      END
```


N=500


N=600



N=800

N=900

N=1000

PROGRAM No. 9

Name: KOLSMI

Purpose: The KOLMOROGOV-SMIRNOV TEST for uniform distribution.

PROGRAM KOLSMI CDC 6600 FTN V3.0-

```
      PROGRAM KOLSMI (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTP
      DIMENSION X(1000)
      COMMON/MNP/FCT
      FCT=2.**(-48)
    5 N=1000
      ST=UNI(1234567)
      DO101=1,1000
   10 X(I)=UNI(0)
      CALL KSTEST(X,N,P1,P2,P3,P4,P5)
   10 WRITE(61,11)P1,P2,P3,P4,P5
   11 FORMAT(3X,E16.7)
      STOP
      END
```

```
.2517357E+00
.5358540E+00
.1625393E+00
.1317679E+00
.1847398E+00
```

```
      SUBROUTINE KSTEST(X,N,P1,P2,P3,P4,P5)
      DIMENSION X(1)
      G(Y)=EXP(CM1*1.2337/(8*Y*Y*CM1+9.86904))
    5 N=1000
      J=0
      CM1=-1
      DM=0
      DP=0
   10 S=2
      P5=0.
      P3=1.
      T=N
      CM=1/(12*T)
   15 CALL ASORT(X,N)
      X(1)=AMAX1(X(1),.1E-8)
      X(N)=AMIN1(X(N),.9999999)
      DO 2 I=1,N
      CM=CM+(X(I)-(I+I-1)/(T+T))**2
   20 CM1=CM1+ALOG(1-X(I))*(I+I-T-T-1)/(T*T)-ALOG(X(I))*(I+I-1)/(T*T)
      G1=I/T-X(I)
      G2=X(I)-(I-1)/T
      IF(G1-DP)4,4,3
    3 DP=G1
   25 4 IF(G2-DM)2,2,5
    5 DM=G2
    2 CONTINUE
      P1=1-EXP(-2*DP*DP*T)
      P2=1-EXP(-2*DM*DM*T)
   30 B=AMAX1(DP,DM)
   33 B=-2*B*B*T
      DO44I=1,50
      S=-S
      T=EXP(I*I*B)
   35 IF(T-.000001)55,55,44
   44 P3=P3+S*T
   55 IF(CM.LE..045) GO TO 17
      IF(CM.LE..19) GO TO 27
      P4=1.-.556852*EXP(-5.43369*CM)-.6789*EXP(-10.8558*CM)
   40 GO TO 15
   17 P4=1.59577*EXP(-.125/CM)
      GO TO 15
   27 P4=(-17.40024*CM+8.35726)*CM-.24994
   15 CM1=CM1*N
   45 IF(CM1.GE.6) GO TO 14
   10 P5=(G(.245341)*.260793+G(.737473)*.16174+G(1.23407)*.06151+
     BG(1.73854)*.014+G(2.25497)*.00183+G(2.7888)*.000128)*4/
     BSQRT(CM1)*EXP(-1.2337/CM1)
      RETURN
   50 14 P5=1.
   13 RETURN
      END
```

```
      SUBROUTINE ASORT(X,N)
      DIMENSION IU(33),IL(33)
      REAL X(1000),T,TT
    5 N=1000
      M=1
      I=1
      J=N
    5 IF(I.GE.J) GO TO 70
   10 K=I
      IJ=(I+J)/2
      T=X(IJ)
      IF(X(I).LE.T) GO TO 20
      X(IJ)=X(I)
   15 X(I)=T
      T=X(IJ)
   20 L=J
      IF(X(J).GE.T) GO TO 40
      X(IJ)=X(J)
   20 X(J)=T
      T=X(IJ)
      IF(X(I).LE.T) GO TO 40
      X(IJ)=X(I)
      X(I)=T
   25 T=X(IJ)
      GO TO 40
   30 X(L)=X(K)
      X(K)=TT
   40 L=L-1
   30 IF(X(L).GT.T) GO TO 40
      TT=X(L)
   50 K=K+1
      IF(X(K).LT.T) GO TO 50
      IF(K.LE.L) GO TO 30
   35 IF(L-I.LE.J-K) GO TO 60
      IL(M)=I
      IU(M)=L
      I=K
      M=M+1
   40 GO TO 80
   60 IL(M)=K
      IU(M)=J
      J=L
      M=M+1
   45 GO TO 80
   70 M=M-1
      IF(M.LE.0) RETURN
      I=IL(M)
      J=IU(M)
   50 IF(J-I.GE.11) GO TO 10
      IF(I.EQ.1) GO TO 5
      I=I-1
   90 I=I+1
      IF(I.EQ.J) GO TO 70
   55 T=X(I+1)
      IF(X(I).LE.T) GO TO 90
      K=I
   95 X(K+1)=X(K)
      K=K-1
   60 IF(T.LT.X(K)) GO TO 95
      X(K+1)=T
      GO TO 90
      END
```

PROGRAM No. 10

Name: RANVAR 1

Purpose: Generation and testing of a random variable.

```
PROGRAM     RANVAR1                                    CDC 6600 FTN V3.0-P

            PROGRAM RANVAR1 (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTP
            DIMENSION X(1000),REXP(1)
            COMMON/MNP/FCT
            FCT=2.**(-48)
 5          ST=UNI(1234567)
            DO10I=1,1000
            T=-ALOG(UNI(0))
         10 X(I)=1.-EXP(-T)
            CALL KSTEST(X,N,P1,P2,P3,P4,P5)
10          WRITE(61,11)P1,P2,P3,P4,P5
         11 FORMAT(3X,E16.7)
            STOP
            END
```

```
 .5358540E+00
 .2517357E+00
 .1625393E+00
 .1317679E+00
 .1847398E+00
```

PROGRAM No. 11

Name: RANVAR 2

Purpose: Generation and testing of a random variable.

```
PROGRAM     RANVAR2                                    CDC 6600 FTN V3,B-P:

            PROGRAM RANVAR2 (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
            DIMENSION X(1000)
            COMMON/MNP/FCT
            FCT=2.**(-48)
      5     ST=UNI(1234567)
            DO10I=1,1000
            T=EXP(-UNI(0))
         10 X(I)=-ALOG(T)
            CALL KSTEST(X,N,P1,P2,P3,P4,P5)
     10     WRITE(61,11)P1,P2,P3,P4,P5
         11 FORMAT(3X,E16.7)
            STOP
            END
```

```
            .2517357E+00
            .5358540E+00
            .1625393E+00
            .1317679E+00
            .1847398E+00
```

PROGRAM No. 12

Name: RANNOR

Purpose: Program to generate a Random Variable with normal density using the Canchy density to which the Kolmorogov-Smirnov test is applied.

```
PROGRAM    RANNOR                                          CDC 6600 FTN V3.0-
      PROGRAM RANNOR (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTP
      DIMENSION X(1000)
      COMMON/MNP/FCT
      FCT=2.**(-48)
      ST=IINI(1234567)
      DO10I=1,1000
      T=RNOR(0)
   10 X(I)=0.5+0.5*ERF(0.707107*T)
      CALL KSTEST(X,N,P1,P2,P3,P4,P5)
      WRITE(61,11)P1,P2,P3,P4,P5
   11 FORMAT(3X,E16.7)
      STOP
      END
```

```
 .1114683E+00
 .7139278E+00
 .4412246E+00
 .6534701E+00
 .5284741E+00
```

```
      FUNCTION RNOR(JJ)
      PHI(X)=EXP(-.5*X*X)/SQRT(2.*PI)
      PI=3.1415926
      C=SQRT(2./PI)
    1 X=CAUCHY(0)
      IF(UNI(0)/(1.+X**2)*PI.GT.C*PHI(X)) GO TO 1
      RNOR=X
      RETURN
      END
```

```
      FUNCTION CAUCHY(JJ)
      COMMON/MNP/FCT
      CAUCHY=TAN(3.1415926*(UNI(0)-0.5))
      RETURN
      END
```

```
      FUNCTION ERF(Y)
      DIMENSION Z(100)
      W=ABS(Y)
      ERF=Y/W
      IF(W.GE.10.)RETURN
      SUM=1.+EXP(-Y**2)
      S=1.
      DO 20 I=1,99
      Z(I)=I*Y/100.
      SUM=SUM+(3.+S)*EXP(-Z(I)**2)
   20 S=-S
      ERF=Y*SUM/(SQRT(3.1415926)*150.)
      RETURN
      END
```

PROGRAM No. 13

Name: RANNOR

Purpose: Program to generate a Random Variable with normal density using polar coordinates.

PROGRAM KANNOR. CDC 6600 FTN V3.0-P296 OPT=1

```
      PROGRAM KANNOR (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPUT)
      DIMENSION X(1000)
      COMMON/MNP/FCT
      FCT=2.**(-48)
      ST=UNI(1234567)
      WRITE(61,30)
   30 FORMAT(1H1)
      WRITE(61,31)
   31 FORMAT(2X* P1      P2      P3      P4      P5  *)
      DO 12 K=1,10
      DO101I=1,1000
      T=RNOR(0)
   10 X(I)=0.5+0.5*ERF(0.707107*T)
      CALL KSTEST(X,N,P1,P2,P3,P4,P5)
      WRITE(61,11)P1,P2,P3,P4,P5
   11 FORMAT(2X,F6.4,2X,F6.4,2X,F6.4,2X,F6.4,2X,F6.4)
      IF(P1.LT.0.05)GO TO 43
      IF(P2.LT.0.05)GO TO 43
      IF(P3.LT.0.05)GO TO 43
      IF(P4.LT.0.05)GO TO 43
      IF(P5.LT.0.05)GO TO 43
      IF(P1.GT.0.95)GO TO 43
      IF(P2.GT.0.95)GO TO 43
      IF(P3.GT.0.95)GO TO 43
      IF(P4.GT.0.95)GO TO 43
      IF(P5.GT.0.95)GO TO 43
      WRITE(61,41)
   41 FORMAT(2X*                                    KSTEST ACCEPTED*)
      GO TO 12
   43 WRITE(61,42)
   42 FORMAT(2X*                                    KSTEST REJECTED*)
   12 CONTINUE
      STOP
      END
```

FUNCTION RNOR CDC 6600 FTN V3.0-

```
      FUNCTION RNOR(JJ)
      LOGICAL FLIP
      DATA FLIP /.TRUE./
      IF(FLIP) GO TO 3
    5 FLIP=.TRUE.
      RNOR=S
      RETURN
    3 R=SQRT(-2.*ALOG(UNI(0)))
      T=6.283186*UNI(0)
   10 RNOR=R*COS(T)
      S=R*SIN(T)
      FLIP=.FALSE.
      RETURN
      END
```

P1	P2	P3	P4	P5	
.4283	.3273	.0574	.1435	.2050	KSTEST ACCEPTED
.2175	.7240	.4596	.4252	.5311	KSTEST ACCEPTED
.0976	.8223	.6466	.5923	.6123	KSTEST ACCEPTED
.4745	.5230	.1469	.3192	.2965	KSTEST ACCEPTED
.5268	.8476	.6962	.6454	.7109	KSTEST ACCEPTED
.2543	.7338	.4777	.4393	.6228	KSTEST ACCEPTED
.4938	.6363	.3073	.2740	.3094	KSTEST ACCEPTED
.8705	.3198	.7415	.5023	.4831	KSTEST ACCEPTED
.0920	.9856	.9712	.9868	.9876	KSTEST REJECTED
.2282	.8054	.6137	.4706	.4216	KSTEST ACCEPTED

PROGRAM No. 14

Name: RANNOR

Purpose: Program to generate a Random Variable with normal density using a one-line generator.

```
PROGRAM      RANNOR                                    CDC 6600 FTN V3.0-P296 OPT=1

             PROGRAM RANNOR (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPUT)
             DIMENSION X(1000)
             COMMON/MNP/FCT
             FCT=2.**(-48)
 5           ST=UNI(1234567)
             WRITE(61,30)
          30 FORMAT(1H1)
             WRITE(61,31)
          31 FORMAT(2X*  P1       P2       P3       P4       P5   *)
10           DO 12 K=1,10
             DO10I=1,1000
             T=RNOR(0)
          10 X(I)=0.5+0.5*ERF(0.707107*T)
             CALL KSTEST(X,N,P1,P2,P3,P4,P5)
15           WRITE(61,11)P1,P2,P3,P4,P5
          11 FORMAT(2X,F6.4,2X,F6.4,2X,F6.4,2X,F6.4,2X,F6.4)
             IF(P1.LT.0.05)GO TO 43
             IF(P2.LT.0.05)GO TO 43
             IF(P3.LT.0.05)GO TO 43
20           IF(P4.LT.0.05)GO TO 43
             IF(P5.LT.0.05)GO TO 43
             IF(P1.GT.0.95)GO TO 43
             IF(P2.GT.0.95)GO TO 43
             IF(P3.GT.0.95)GO TO 43
25           IF(P4.GT.0.95)GO TO 43
             IF(P5.GT.0.95)GO TO 43
             WRITE(61,41)
          41 FORMAT(2X*                                          KSTEST ACCEPTED*)
             GO TO 12
30        43 WRITE(61,42)
          42 FORMAT(2X*                                          KSTEST REJECTED*)
          12 CONTINUE
             STOP
             END
```

FUNCTION RNOR CDC 6600 FTN V3.0-P

```
      FUNCTION RNOR(JJ)
      COMMON/MNP/FCT
      RNOR=SQRT(-2.*ALOG(UNI(0)))*COS(3.1415926*UNI(0))
      RETURN
5     END
```

P1	P2	P3	P4	P5	
.4710	.4357	.0922	.1308	.1127	KSTEST ACCEPTED
.2655	.2616	.0021	.0126	.0587	KSTEST REJECTED
.3594	.8800	.7604	.7016	.7674	KSTEST ACCEPTED
.6930	.4244	.4036	.4139	.3646	KSTEST ACCEPTED
.1814	.9322	.8644	.7670	.7507	KSTEST ACCEPTED
.0880	.7259	.4630	.4323	.3345	KSTEST ACCEPTED
.5700	.5402	.2074	.2162	.1548	KSTEST ACCEPTED
.3119	.5623	.1969	.2258	.1733	KSTEST ACCEPTED
.7401	.4944	.4893	.6076	.5027	KSTEST ACCEPTED
.3207	.2612	.0097	.0148	.0141	KSTEST REJECTED

PROGRAM No. 15

Name: DICE

Purpose: Use of random number sequence to simulate the throwing of two die.

PROGRAM DICE CDC 6600 FTN V3.0-P

```
      PROGRAM DICE (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPUT)
      DIMENSION PRDNS(12),PRDST(12),IC(12)
      COMMON/MNP/FCT
      FCT=2.**(-48)
    5 ST=UNI(1234567)
      DO 3 J=1,12
    3 IC(J)=0
      DO 1 M=1,1000
      I=ITHROW(0)
   10 IC(I)=IC(I)+1
    1 CONTINUE
      PRDNS(1)=0.
      PRDST(1)=0.
      DO 2 I=2,12
   15 PRDNS(I)=IC(I)*0.001
      PRDST(I)=PRDST(I-1)+PRDNS(I)
    2 WRITE(61,11)I,PRDNS(I),PRDST(I)
   11 FORMAT(3X,I2,3X,F6.4,3X,F6.4)
      STOP
   20 END


      FUNCTION ITHROW(JJ)
      COMMON/MNP/FCT
      I=6.*UNI(0)+1
    5 J=6.*UNI(0)+1
      ITHROW=I+J
      RETURN
      END
```

```
 2    .0330    .0330
 3    .0470    .0800
 4    .0860    .1660
 5    .1130    .2790
 6    .1390    .4180
 7    .1700    .5880
 8    .1450    .7330
 9    .0970    .8300
10    .0760    .9060
11    .0570    .9630
12    .0370   1.0000
```

PROGRAM No. 16

Name: MATQ2

Purpose: Use of random number sequence to determine the probability of events from a given probability matrix.

```
PROGRAM     MATQ2                                          CDC 6600 FTN V3.0-P

      PROGRAM MATQ2 (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPUT
      DIMENSION Q(3,3),ICOUNT(3),COUNT(3)
      COMMON/MNP/FCT
      FCT=2.**(-48)
      ST=UNI(1234567)
      DO 20 K=1,3
   20 READ(60,21)(Q(K,L),L=1,3)
   21 FORMAT(3(2X,F4.2))
      DO 22 K=1,3
   22 WRITE(61,23)(Q(K,L),L=1,3)
   23 FORMAT(3(2X,F4.2))
      N=5
      DO 6 J=1,3
    6 ICOUNT(J)=0
      DO 3 M=1,1000
      I=3.*UNI(0)+1
      DO 1 IC=1,N
      I=ISERCH(Q,I,M)
    1 CONTINUE
      ICOUNT(I)=ICOUNT(I)+1
    3 CONTINUE
      DO 7 J=1,3
      COUNT(J)=ICOUNT(J)*0.001
    7 WRITE(61,8)COUNT(J)
    8 FORMAT(3X,E14.5)
      STOP
      END


      FUNCTION ISERCH(Q,I,M)
      DIMENSION Q(3,3)
      Y=UNI(0)
      ISERCH=1
      DO 10 K=1,3
      IF(Y.LT.Q(I,K))RETURN
      ISERCH=ISERCH+1
   10 CONTINUE
      ISERCH=3
      RETURN
      END


  .40   .70  1.00
  .20   .50  1.00
  .40   .60  1.00
      .33200E+00
      .25000E+00
      .40000E+00
```

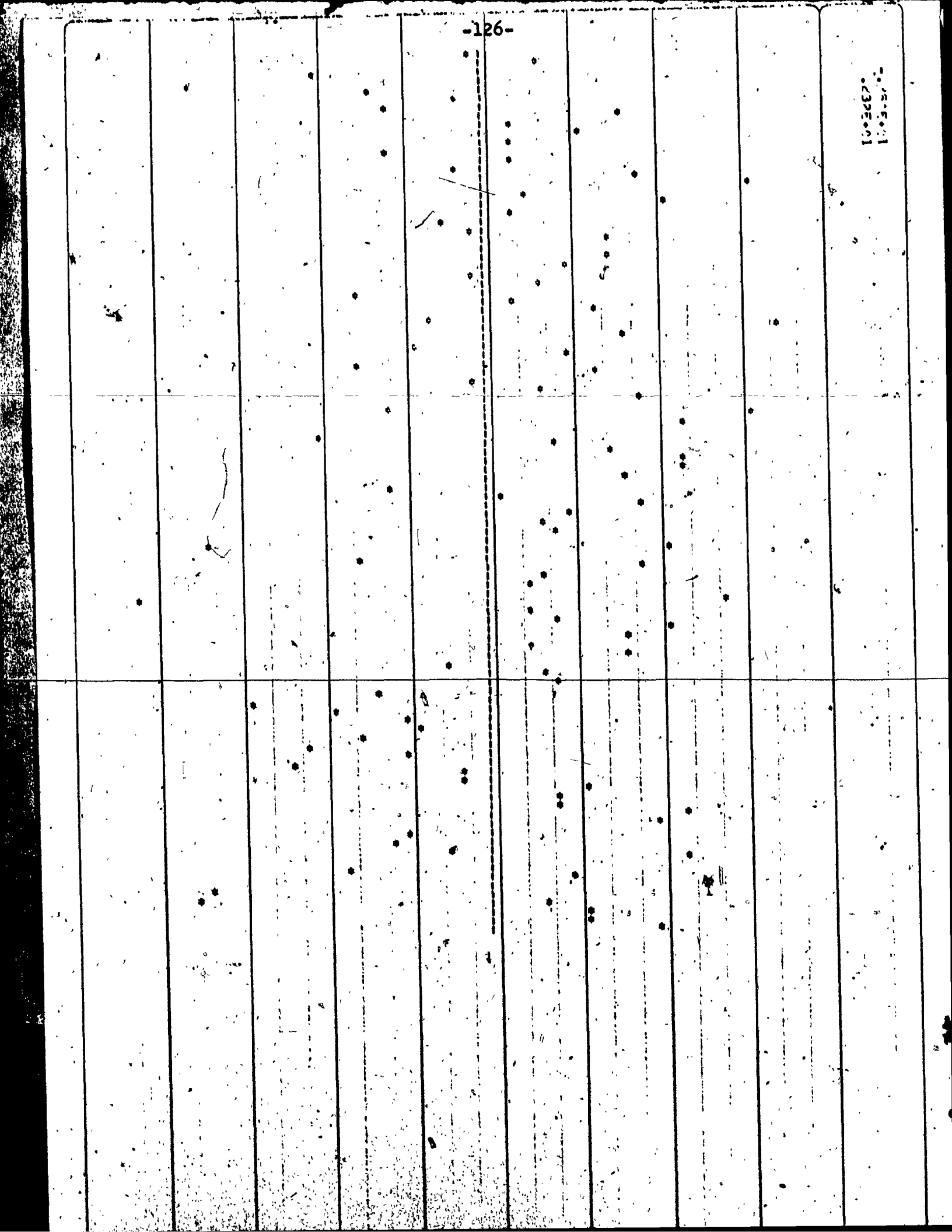
PROGRAM No. 17

Name: PLRNOR

Purpose: Program to obtain a graphical representation of ten sets of one hundred uniform intervals of white noise.

```
      PROGRAM    PLRNOR                                        CDC 6600 FTN V3.0
                       PROGRAM PLRNOR (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUT
                       DIMENSION X(100),T(100),M(100,54)
                       INTEGER BLANK,STAR,ORD
                       COMMON/MNP/FCT
     5                 COMMON/MNQ/BLANK,STAR,ORD
                       DATA BLANK,STAR,ORD/1H ,1H*,1H=/
                       FCT=2.**(-48)
                       ST=SNI(1234567)
                       DO 33 K=1,10
    10                 DO 10 I=1,100
                       T(I)=I*0.01
                    10 X(I)=RNOR(0)
                       CALL PLOT(X,T,M)
                    33 CONTINUE
    15                 STOP
                       END


      SUBROUTINE  PLOT
                       SUBROUTINE PLOT(X,T,M)
                       DIMENSION X(100),T(100),M(100,54)
                       INTEGER BLANK,STAR,ORD
                       COMMON/MNQ/BLANK,STAR,ORD
     5                 XMIN=X(1)
                       XMAX=X(1)
                       DO 20 I=2,100
                       IF(X(I).LT.XMIN)XMIN=X(I)
                       IF(X(I).GT.XMAX)XMAX=X(I)
    10                 20 CONTINUE
                       Z=XMAX-XMIN
                       DO11I=1,100
                       DO11J=1,54
                    11 M(I,J)=BLANK
    15                 IF(XMIN.GT.0.) GO TO 15
                       K=53.*(-XMIN)/Z+1.
                       DO 16 J=1,100
                       IX=I
                       IY=K
    20                 16 M(IX,IY)=ORD
                       15 DO12I=1,100
                       IX=I
                       IY=53.*(X(I)-XMIN)/Z+1.
                    12 M(IX,IY)=STAR
    25                 WRITE(6,30)
                    30 FORMAT(1H1)
                       WRITE(6,32)XMIN,XMAX
                    32 FORMAT(3X,F12.3)
                       DO131=1,54
    30                 13 WRITE(6,14)(M(J,I),J=1,100)
                    14 FORMAT(3X,100A1)
                       RETURN
                       END
```

-.144E+01
.254E+01



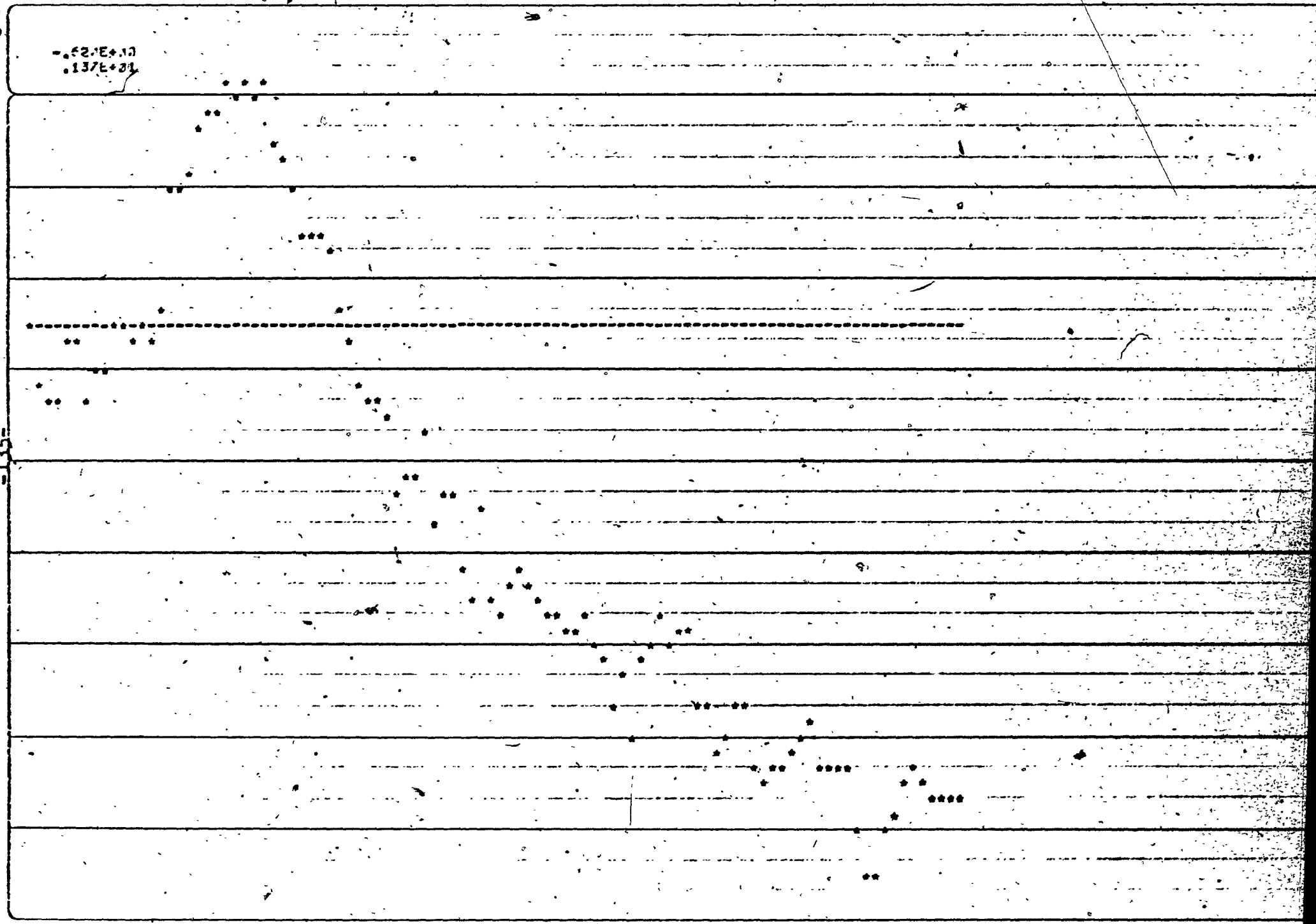
PROGRAM No. 18

Name: STOPRO

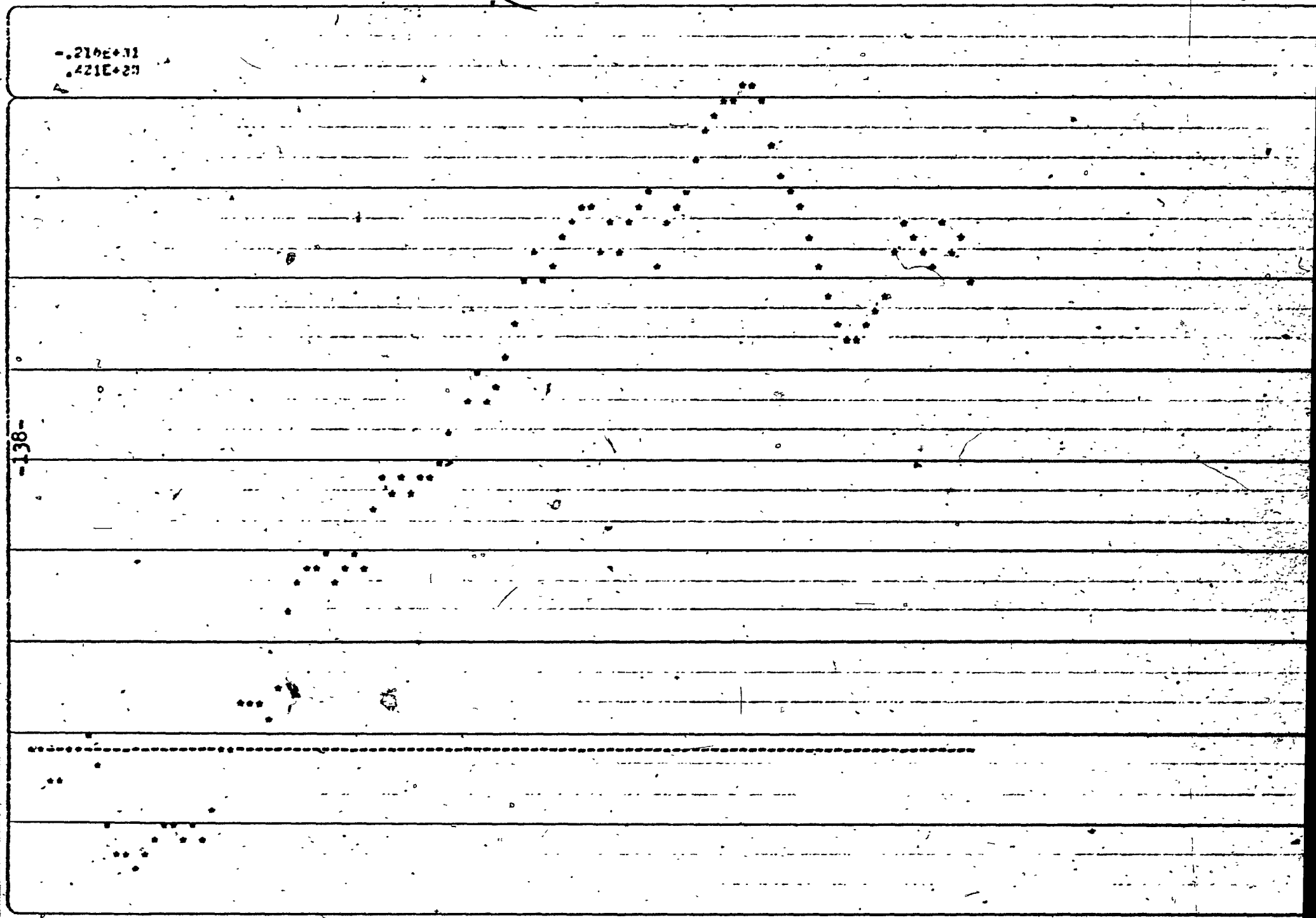
Purpose: Program to obtain a graphical representation of ten sets of one hundred values of a stochastic process.

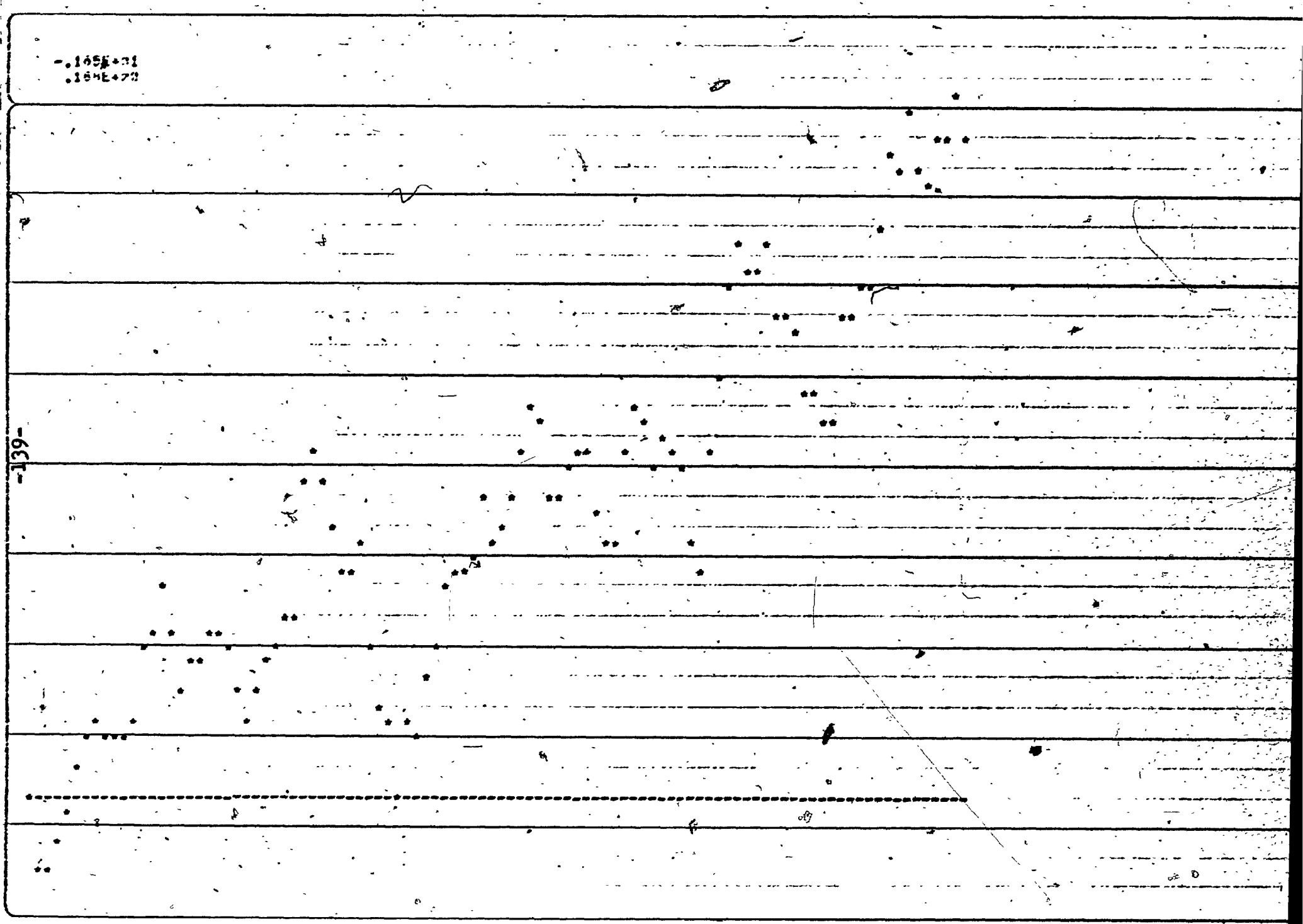
```
PROGRAM   STOPRO                                    CDC 6600 FTN V3.0-P:

          PROGRAM STOPRO (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
          DIMENSIONX(100),T(100),M(100,54)
          INTEGER BLANK,STAR,ORD
          COMMON/MNP/FCT
 5        COMMON/MNQ/BLANK,STAR,ORD
          DATA BLANK,STAR,ORD/1H ,1H*,1H=/
          FCT=2.**(-48)
          ST=UNI(1234567)
          DO 33 K=1,10
10        SIG=0.1
          X(1)=0.
          T(1)=0.01
          DO 10 I=2,100
          T(I)=I*0.01
15     10 X(I)=X(I-1)+SIG*RNOR(0)
          CALL PLOT(X,T,M)
       33 CONTINUE
          STOP
          END
```

-.697E+24
.153E+33

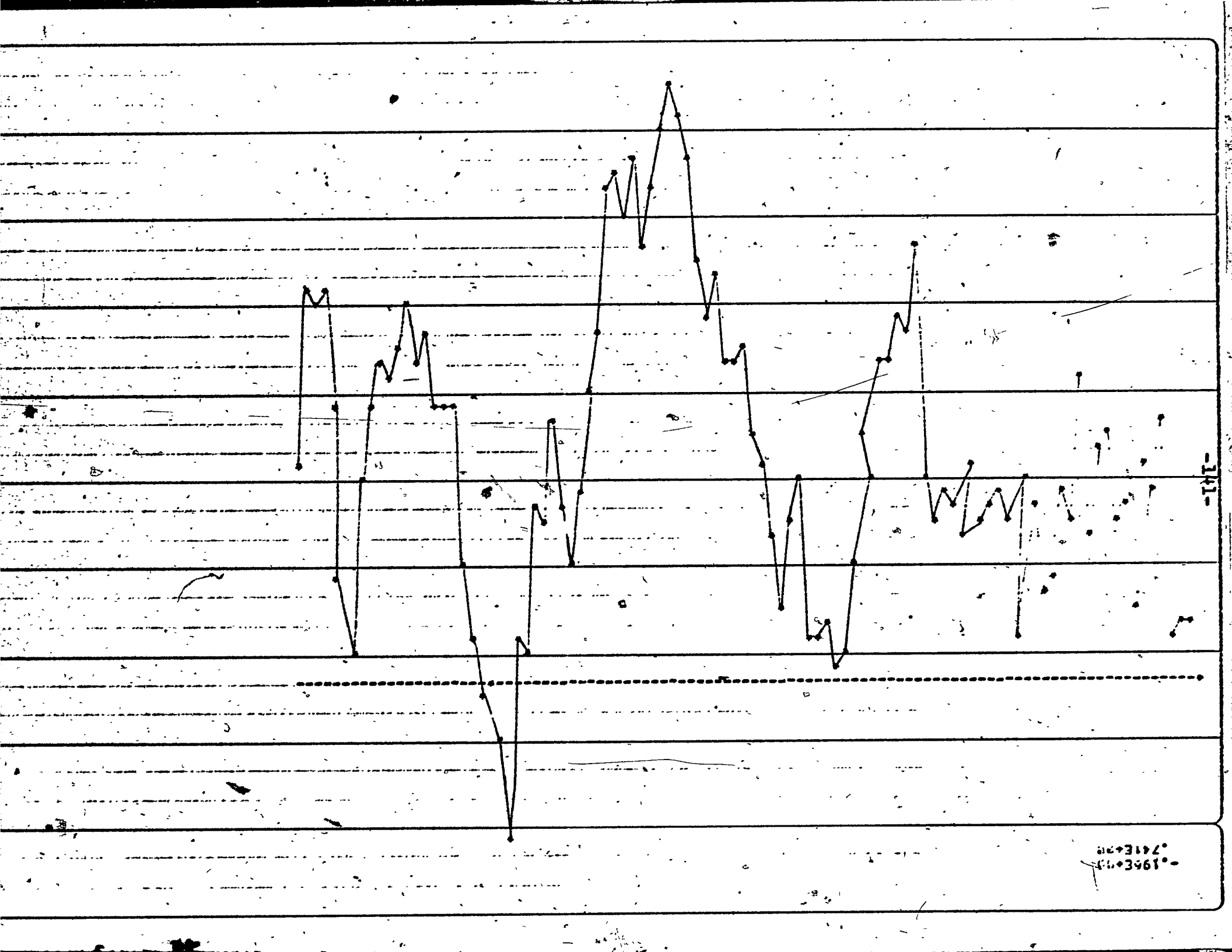


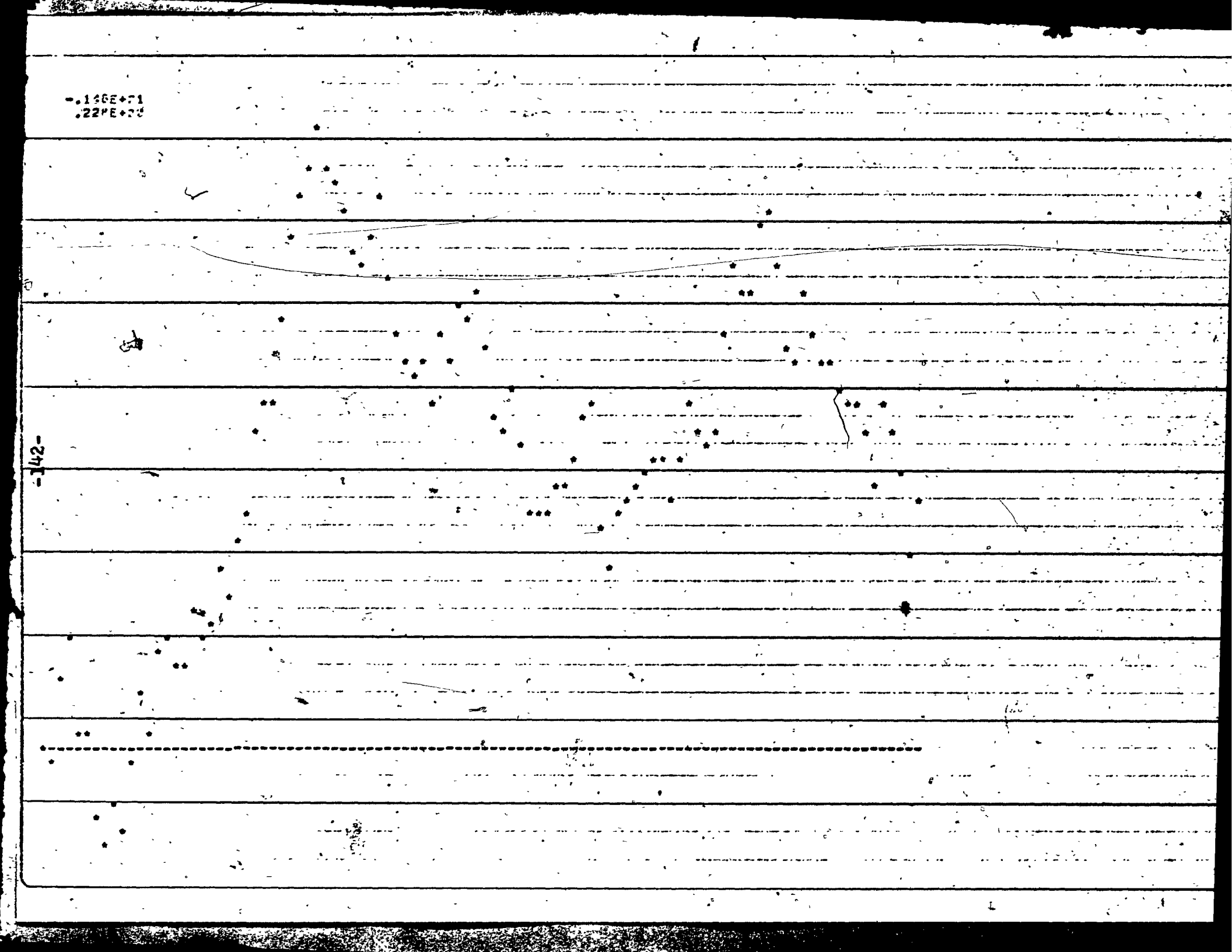
-.155E+03
-.133E+11


-.216E+31
.421E+29


-.165E+01
.164E+02

-.75-E-71
.124E+71




-.146E+71
.22PE+70
-142-

ANNEXURE

PROGRAM No. 19

Name: CRODUR

Purpose: Program to determine the number of excursions of a Stochastic Process above a certain limit.

PROGRAM CRODUR CDC 6600 FTN V3.0-P

```
      PROGRAM CRODUR (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
      DIMENSION IDURN(100)
      COMMON/MNP/FCT
      FCT=2.**(-48)
    5 17 READ(60,50)A
   50 FORMAT(F6.0)
      IF(A.GE.0.6)STOP
      ST=UNI(1234567)
      SUMK=0.
   10 DO 1 IC=1,100
      CALL CROSS(A,K,IDURN)
      SUMK=SUMK+K
    1 CONTINUE
      AVE=SUMK*0.01
   15 WRITE(61,51)A,AVE
   51 FORMAT(3X,E12.3,6X,E16.7)
      GO TO 17
      END


      SUBROUTINE CROSS(A,K,IDURN)
      DIMENSION S(100),IDURN(100)
      SIG=0.1
    5 K=0
      I=1
      S(1)=0.
    1 I=I+1
      IF(I.GT.100)RETURN
   10 S(I)=S(I-1)+SIG*RNOR(0)
      IF((S(I).GE.A).AND.(S(I-1).LT.A))GOTO3
      GOTO1
    3 K=K+1
      J=I
   15 4 I=I+1
      IF(I.GT.100)RETURN
      S(I)=S(I-1)+SIG*RNOR(0)
      IDURN(K)=(I-J)
      IF(S(I).LT.A)GOTO1
   20 GOTO4
      END
```

```
      .200E+00          .2410000E+01
      .250E+00          .2150000E+01
      .300E+00          .2170000E+01
      .350E+00          .1960000E+01
      .400E+00          .2100000E+01
      .450E+00          .1670000E+01
      .500E+00          .1620000E+01
```


PROGRAM No. 20

Name: DURATN

Purpose: Program to determine the duration of excursions of a stochastic process above a certain limit.

```
PROGRAM     DURATN                                        CDC 6600 FTN V3.0-P

            PROGRAM DURATN (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
            DIMENSION IDURN(100),ICDURN(100),PDSDRN(100),PDNDRN(10
            COMMON/MNP/FCT
            FCT=2.**(-48)
      5     ST=UNI(1234567)
            READ(60,50)A
         50 FORMAT(F6.0)
            SUMK=0.
            DO 40 J=1,100
     10  40 ICDURN(J)=0
            DO 1 IC=1,100
            CALL CROSS(A,K,IDURN)
            SUMK=SUMK+K
            IF(K.EQ.0) GO TO 1
     15     DO 2 IDK=1,K
            L=IDURN(K)
         2  ICDURN(L)=ICDURN(L)+1
         1  CONTINUE
            IF(SUMK.LE.0.1) GO TO 15
     20     RSUMK=1./SUMK
            PDNDRN(1)=ICDURN(1)*RSUMK
            PDSDRN(1)=PDNDRN(1)
            DO 3 I=2,100
            PDNDRN(I)=ICDURN(I)*RSUMK
     25  3  PDSDRN(I)=PDSDRN(I-1)+PDNDRN(I)
            DO 4 I=1,100
         4  WRITE(61,11)I,PDNDRN(I),PDSDRN(I)
        11  FORMAT(3X,I3,3X,F6.4,3X,F6.4)
        15  STOP
     30     END
```

SUBROUTINE CROSS CDC 6600 FTN

SYMBOLIC REFERENCE MAP

ENTRY POINTS
2 CROSS

VARIABLES	SN	TYPE		RELOCATION				
0	A	REAL		F.P.	64	I		INT
0	IDURN	INTEGER	ARRAY	F.P.	65	J		INT
0	K	INTEGER		F.P.	66	S		REA
63	SIG	REAL						

EXTERNALS	TYPE	ARGS
RNOR	REAL	1

STATEMENT LABELS
20 1 35 3

STATISTICS
PROGRAM LENGTH 240B 160

1	,1878	,1878
2	,1174	,3052
3	,0423	,3474
4	,0329	,3803
5	,0141	,3944
6	,0188	,4131
7	,0282	,4413
8	0,0000	,4413
9	0,0000	,4413
10	,0282	,4695
11	0,0000	,4695
12	,0329	,5023
13	,0047	,5070
14	,0469	,5540
15	0,0000	,5540
16	,0188	,5728
17	,0376	,6103
18	0,0000	,6103
19	0,0000	,6103
20	0,0000	,6103
21	0,0000	,6103
22	0,0000	,6103
23	,0188	,6291
24	0,0000	,6291
25	0,0000	,6291
26	0,0000	,6291
27	,0141	,6432
28	0,0000	,6432
29	0,0000	,6432
30	,0188	,6620
31	0,0000	,6620
32	,0047	,6667
33	,0047	,6714
34	0,0000	,6714
35	0,0000	,6714
36	,0751	,7465
37	0,0000	,7465
38	,0610	,8075
39	0,0000	,8075
40	0,0000	,8075
41	0,0000	,8075
42	0,0000	,8075
43	0,0000	,8075
44	,0141	,8216

45	0,0000	,8216
46	0,0000	,8216
47	0,0000	,8216
48	,0141	,8357
49	,0188	,8545
50	0,0000	,8545
51	0,0000	,8545
52	,0282	,8826
53	,0094	,8920
54	0,0000	,8920
55	0,0000	,8920
56	0,0000	,8920
57	0,0000	,8920
58	,0141	,9061
59	0,0000	,9061
60	0,0000	,9061
61	,0047	,9108
62	,0094	,9202
63	0,0000	,9202
64	,0047	,9249
65	0,0000	,9249
66	,0141	,9390
67	0,0000	,9390
68	0,0000	,9390
69	0,0000	,9390
70	0,0000	,9390
71	0,0000	,9390
72	0,0000	,9390
73	,0047	,9437
74	0,0000	,9437
75	0,0000	,9437
76	,0047	,9484
77	0,0000	,9484
78	0,0000	,9484
79	,0094	,9577
80	0,0000	,9577
81	,0094	,9671
82	0,0000	,9671
83	0,0000	,9671
84	0,0000	,9671
85	,0094	,9765
86	,0047	,9812
87	0,0000	,9812
88	0,0000	,9812
89	0,0000	,9812
90	,0141	,9953
91	0,0000	,9953
92	0,0000	,9953
93	0,0000	,9953
94	0,0000	,9953
95	,0047	1,0000
96	0,0000	1,0000
97	0,0000	1,0000
98	0,0000	1,0000
99	0,0000	1,0000
100	0,0000	1,0000

PROGRAM No. 21

Name: STPROY

Purpose: Program to generate a Stochastic Process of known statistical properties.

```
      PROGRAM    STPROY                                    CDC 6600 FTN V3.

                 PROGRAM STPROY (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OU
                 DIMENSION A(5050),G(5050),AM(100),X(100),Y(100)
                 COMMON/MNP/FCT
                 FCT=2.**(-48)
    5            IDMN=5050
                 BETA=0.1
                 N=100
                 ST=UNI(1234567)
                 DO 20 I=1,N
   10            DO 20 J=1,I
                 K=(I*(I-1))/2+J
              20 A(K)=EXP(-BETA*(I-J)**2)
                 DO 21 I=1,N
              21 AM(I)=0.0
   15            CALL DECOMP(A,G,N,IDMN)
                 DO 2 I=1,N
              2  X(I)=RNOR(0)
                 DO 3 I=1,N
                 SM=AM(I)
   20            LI=(I*(I-1))/2
                 DO 4 J=1,I
              4  SM=SM+G(LI+J)*X(J)
              3  Y(I)=SM
                 WRITE(61,30)
   25         30 FORMAT(1H1)
                 WRITE(61,22) (Y(I),I=1,N)
              22 FORMAT(3X,E16.7)
                 STOP
                 END
```

SUBROUTINE DECOMP CDC 6600 FTN V3.(

```
      SUBROUTINE DECOMP(A,G,N,ID,N)
      DIMENSION A(1DMN),G(1DMN)
      DO 1 J=1,N
      KJ=(J*(J+1))/2
 5    LJ=(J*(J-1))/2
      SM=A(KJ)
      IF(J.EQ.1)GO TO 2
      MAXK=J-1
      DO 3 K=1,MAXK
10    SM=SM-G(LJ+K)**2
    3 CONTINUE
      IF(SM.LE.0.) GO TO 20
    2 G(KJ)=SQRT(SM)
      MINI=J+1
15    IF(J.EQ.N)GO TO 7
      DO 5 I=MINI,N
      LI=(I*(I-1))/2
    4 SM=A(LI+J)
      MAXL=J-1
20    IF(MAXL.EQ.0)GO TO 4
      DO 6 L=1,MAXL
      SM=SM-G(LI+L)*G(LJ+L)
    6 CONTINUE
    4 G(LI+J)=SM/G(KJ)
25  5 CONTINUE
    1 CONTINUE
    7 RETURN
   20 WRITE(61,21)N,J,K,SM
   21 FORMAT(3X,I3,3X,I3,3X,I3,3X,E16.7)
30    STOP
      END
```

-.2501628E+00
-.5637448E+00
-.6840223E+00
-.9154083E+00
-.1341296E+01
-.1562975E+01
-.1265495E+01
-.5525174E+00
.2789565E+00
.9667325E+00
.1236223E+01
.8129881E+00
-.1642523E+00
-.9242547E+00
-.7661772E+00
.8554851E-01
.7339243E+00
.5982799E+00
-.1993965E+00
-.1183687E+01
-.1766869E+01
-.1444945E+01
-.3630494E+00
.5886145E+00
.6782654E+00
.7425410E-01
-.4306037E+00
-.2751163E+00
.3406828E+00
.7867497E+00
.7259224E+00
.4011526E+00
.2358217E+00
.3230087E+00
.4339260E+00
.2830704E+00
-.1977608E+00
-.6454025E+00
-.6389128E+00
-.2573364E+00
.1066213E+00
.1710389E+00
-.1399940E+00
-.6209542E+00
-.8847960E+00
-.7435090E+00
-.3114186E+00
.3052020E+00
.1103333E+01
.1837864E+01
.2032542E+01
.1568483E+01
.8519157E+00
.3740022E+00
.3351275E+00
.5218202E+00
.5828729E+00
.4492659E+00
.1767054E+00
-.3374591E+00
-.1038850E+01

-.1125540E+01
-.5919872E+00
-.3430852E+00
-.4100626E+00
-.4563481E+00
-.3326096E+00
-.1647767E+00
.6954734E-01
.4392978E+00
.6585392E+00
.3559752E+00
-.3181305E+00
-.8112540E+00
-.9250795E+00
-.9288289E+00
-.8937497E+00
-.6261774E+00
-.2118385E+00
-.3602778E-01
-.2836118E+00
-.7241007E+00
-.1065272E+01
-.1207928E+01
-.1010240E+01
-.3588534E+00
.5164177E+00
.1086575E+01
.1121164E+01
.9167840E+00
.8854516E+00
.1083294E+01
.1192366E+01
.8821127E+00
.1438342E+00
-.6894694E+00
-.1225640E+01
-.1260476E+01

PROGRAM No. 22

Name: GMAT 1

Purpose: Program to check subroutine for determining G, such that GG^T equals A.

```
PROGRAM    GMAT1                                          CDC 6600 FTN V3.0-

           PROGRAM GMAT1 (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
           DIMENSION A(4,4),G(4,4)
           N=4
           READ(60,10)((A(I,J),J=1,N),I=1,N)
 5       10 FORMAT(4F6.0)
           DO 1 J=1,N
           SM=A(J,J)
           IF(J.EQ.1)GO TO 2
           MAXK=J-1
10         DO 3 K=1,MAXK
           SM=SM-G(J,K)**2
         3 CONTINUE
         2 G(J,J)=SQRT(SM)
           MINI=J+1
15         IF(J.EQ.N)GO TO 7
           DO 5 I=MINI,N
           SM=A(I,J)
           MAXL=J-1
           IF(MAXL.EQ.0)GO TO 4
20         DO 6 L=1,MAXL
           SM=SM-G(I,L)*G(J,L)
         6 CONTINUE
         4 G(I,J)=SM/G(J,J)
         5 CONTINUE
25       1 CONTINUE
         7 DO 8 I=1,N
           MAXJ=I
           WRITE(61,16)(G(I,J),J=1,MAXJ)
         8 CONTINUE
30      16 FORMAT(4F12.7)
           DO 11 I=1,N
           DO 11 J=1,N
           SUM=0.
           DO 12 K=1,N
35      12 SUM=SUM+G(I,K)*G(J,K)
        11 WRITE(61,17)SUM,A(I,J)
        17 FORMAT(3X,F12.7,3X,F12.7)
           STOP
           END
```

```
2.0000000
 .5000000   2.1794495
 .5000000    .3441236   2.3730948
 .5000000    .3441236    .2661415   2.5613956
  4.0000000     4.0000000
  1.0000000     1.0000000
  1.0000000     1.0000000
  1.0000000     1.0000000
  1.0000000     1.0000000
  5.0000000     5.0000000
  1.0000000     1.0000000
  1.0000000     1.0000000
  1.0000000     1.0000000
  1.0000000     1.0000000
  6.0000000     6.0000000
  1.0000000     1.0000000
  1.0000000     1.0000000
  1.0000000     1.0000000
  1.0000000     1.0000000
```

SIR GEOR

PROGRAM No. 23

Name: STOCHY

Purpose: Program to obtain values of a stochastic process, a plot of these values and a count of the number of crossings above a limit, their duration, probability density and distribution function.

```
PROGRAM    STOCHY                                        CDC 6600 FTN V3.(
                    PROGRAM STOCHY (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUT
                    DIMENSION A(5050),G(5050),AM(100),X(100),T(100),Y(1(
                   QIDURN(100),ICDURN(100),PDSDRN(100),PDNDRN(100),
                   QIDURT(100),ICDURT(100),PDSDRT(100),PDNDRT(100)
      5             INTEGER BLANK,STAR,ORD
                    COMMON/MNP/FCT
                    COMMON/MNQ/BLANK,STAR,ORD
                    DATA BLANK,STAR,ORD/1H ,1H*,1H-/
                 17 READ(60,50)B
     10             50 FORMAT(F6.0)
                    IF(B.LT.-0.1)GO TO 19
                    WRITE(61,30)
                 30 FORMAT(1H1)
                    FCT=2.**(-48)
     15             N=100
                    IDMN=5050
                    BETA=0.1
                    ST=UNI(1234567)
                    DO 20 I=1,N
     20             DO 20 J=1,I
                    K=(I*(I-1))/2+J
                 20 A(K)=EXP(-BETA*(I-J)**2)
                    DO 21 I=1,N
                 21 AM(I)=0.0
     25             CALL DECOMP(A,G,N,IDMN)
                    DO 2 I=1,N
                    T(I)=I*0.01
                  2 X(I)=RNOR(0)
                    DO 3 I=1,N
     30             SM=AM(I)
                    LI=(I*(I-1))/2
                    DO 4 J=1,I
                  4 SM=SM+G(LI+J)*X(J)
                  3 Y(I)=SM
     35             WRITE(61,23)
                 23 FORMAT(3X,*    Y(I)      *)
                    WRITE(61,22)(Y(I),I=1,N)
                 22 FORMAT(3X,E16.7)
                    CALL PLOT(Y,T,M,N)
     40             CALL CRODUR(Y,T,B,N,IDURN,ICDURN,PDSDRN,PDNDRN,
                   QIDURT,ICDURT,PDSDRT,PDNDRT)
                    GO TO 17
                 19 STOP
                    END
```

SUBROUTINE CRODUR CDC 6600 FTN V3.0-P296 OPT=1

```
      SUBROUTINE CRODUR(Y,T,B,N,IDURN,ICDURN,PDSDRN,PDNDRN,
     QIDURT,ICDURT,PDSDRT,PDNDRT)
      DIMENSION Y(100),IDURN(100),ICDURN(100),PDSDRN(100),PDNDRN(100),
     QIDURT(100),ICDURT(100),PDSDRT(100),PDNDRT(100)
      COMMON/MNP/FCT
      SUMK=0.
      SUMM=0.
      DO 40 J=1,100
      ICDURT(J)=0
   40 ICDURN(J)=0
      CALL CROSS(Y,B,K,N,IDURN,M,IDURT)
      SUMK=SUMK+K
      IF(K.EQ.0) GO TO 1
      DO 2 IDK=1,K
      L=IDURN(IDK)
    2 ICDURN(L)=ICDURN(L)+1
      WRITE(61,92)
   92 FORMAT(3X*       B                        SUMK       *)
    1 WRITE(61,51)B,SUMK
   51 FORMAT(3X,E12.3,6X,E16.7)
      IF(SUMK.LE.0.1) GO TO 95
      WRITE(61,30)
   30 FORMAT(1H1)
      WRITE(61,82)
   82 FORMAT(3X*  I       PDNDRN(I)      PDSDRN(I)  *)
      RSUMK=1./SUMK
      PDNDRN(1)=ICDURN(1)*RSUMK
      PDSDRN(1)=PDNDRN(1)
      I=2
    5 PDNDRN(I)=ICDURN(I)*RSUMK
      PDSDRN(I)=PDSDRN(I-1)+PDNDRN(I)
      IF(PDSDRN(I).GE.1.)GO TO 15
      IF(I.GE.100)GO TO 15
      I=I+1
      GO TO 5
   15 WRITE(61,11)(J,PDNDRN(J),PDSDRN(J),J=1,I)
   11 FORMAT(3X,I3,7X,F6.4,7X,F6.4)
   95 SUMM=SUMM+M
      IF(M.EQ.0) GO TO 71
      DO 72 IDM=1,M
      L=IDURT(IDM)
   72 ICDURT(L)=ICDURT(L)+1
      WRITE(61,73)
   73 FORMAT(3X*       B                        SUMM       *)
   71 WRITE(61,74)B,SUMM
   74 FORMAT(3X,E12.3,6X,E16.7)
      IF(SUMM.LE.0.1) GO TO 96
      WRITE(61,30)
      WRITE(61,75)
   75 FORMAT(3X*  I       PDNDRT(I)      PDSDRT(I)  *)
      RSUMM=1./SUMM
      PDNDRT(1)=ICDURT(1)*RSUMM
      PDSDRT(1)=PDNDRT(1)
      I=2
   85 PDNDRT(I)=ICDURT(I)*RSUMM
```


```
SUBROUTINE  CRODUR                                    CDC 6600 FTN V3.0-P296 OPT=1

                PDSDRT(I)=PDSDRT(I-1)+PDNDRT(I)
                IF(PDSDRT(I).GE.1.)GO TO 25
                IF(I.GE.100)GO TO 25
                I=I+1
60              GO TO 85
25              WRITE(61,11)(J,PDNDRT(J),PDSDRT(J),J=1,I)
96              RETURN
                END
```

Y(I)	
-.2501628E+00	
-.5637448E+00	-.1344417E+01
-.6840223E+00	-.1555119E+01
-.9154083E+00	-.1125560E+01
-.1341296E+01	-.5919872E+00
-.1562975E+01	-.3430852E+00
-.1265495E+01	-.4100626E+00
-.5525174E+00	-.4563481E+00
.2789565E+00	-.3326096E+00
.9667325E+00	-.1647767E+00
.1236223E+01	.6954734E-01
.8129881E+00	.4392978E+00
-.1642523E+00	.6585392E+00
-.9242547E+00	.3559752E+00
-.7661772E+00	-.3181305E+00
.8554851E-01	-.8112540E+00
.7339243E+00	-.9250795E+00
.5982799E+00	-.9288289E+00
-.1993965E+00	-.8937497E+00
-.1183687E+01	-.6261774E+00
-.1766869E+01	-.2118385E+00
-.1444945E+01	-.3602778E-01
-.3630494E+00	-.2836118E+00
.5886145E+00	-.7241007E+00
.6782654E+00	-.1065272E+01
.7425410E-01	-.1201928E+01
-.4306037E+00	-.1010240E+01
-.2751163E+00	-.3586534E+00
.3406828E+00	.5164177E+00
.7867497E+00	.1086575E+01
.7259224E+00	.1121164E+01
.4011526E+00	.9167840E+00
.2358217E+00	.8854506E+00
.3230087E+00	.1083294E+01
.4339260E+00	.1192366E+01
.2830704E+00	.8821127E+00
-.1977608E+00	.1438342E+00
-.6454025E+00	-.6894694E+00
-.6389128E+00	-.1225640E+01
-.2573364E+00	-.1260476E+01
.1066213E+00	
.1710389E+00	
-.1399940E+00	
-.6209542E+00	
-.8847960E+00	
-.7435090E+00	
-.3114186E+00	
.3052020E+00	
.1103333E+01	
.1837864E+01	
.2032542E+01	
.1568483E+01	
.8519157E+00	
.3740022E+00	
.3351275E+00	
.5218202E+00	
.5828729E+00	
.4492659E+00	
.1767054E+00	
-.3374591E+00	

B	SUMK
.400E+00	.9000000E+01

I	PDNDRN(I)	PDSDRN(I)
1	.1111	.1111
2	.3333	.4444
3	.3333	.7778
4	0.0000	.7778
5	.1111	.8889
6	0.0000	.8889
7	0.0000	.8889
8	.1111	1.0000

B	SUMM
.400E+00	.1000000E+02

I	PDNDRT(I)	PDSDRT(I)
1	0.0000	0.0000
2	.2000	.2000
3	.1000	.3000
4	.2000	.5000
5	.1000	.6000
6	0.0000	.6000
7	0.0000	.6000
8	0.0000	.6000
9	0.0000	.6000
10	.1000	.7000
11	0.0000	.7000
12	0.0000	.7000
13	.2000	.9000
14	0.0000	.9000
15	.1000	1.0000

PROGRAM No. 24

Name: STOCHY

Purpose: Program to provide statistical data on the crossings of a stochastic process above a certain limit.

PROGRAM STOCHY CDC 6600 FTN V3.0-P296 OPT=1

```
      PROGRAM STOCHY (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPUT)
      DIMENSION A(5050),G(5050),AM(100),X(100),T(100),Y(100),M(100,54),
     QIDURN(100),ICDURN(100),PDSDRN(100),PDNDRN(100),
     QIDURT(100),ICDURT(100),PDSDRT(100),PDNDRT(100)
      INTEGER BLANK,STAR,ORD
      COMMON/MNP/FCT
      COMMON/MNQ/BLANK,STAR,ORD
      DATA BLANK,STAR,ORD/1H ,1H*,1H=/
   17 READ(60,50)B
   50 FORMAT(F6.0)
      IF(B.LT.-0.1)GO TO 19
      WRITE(61,30)
   30 FORMAT(1H1)
      FCT=2.**(-48)
      N=100
      IDMN=5050
      BETA=0.1
      ST=UNI(1234567)
      DO 20 I=1,N
      DO 20 J=1,I
      K=(I*(I-1))/2+J
   20 A(K)=EXP(-BETA*(I-J)**2)
      DO 21 I=1,N
   21 AM(I)=0.0
      CALL DECOMP(A,G,N,IDMN)
      DO 2 I=1,N
      T(I)=I*0.01
    2 X(I)=RNOR(0)
      DO 3 I=1,N
      SM=AM(I)
      LI=(I*(I-1))/2
      DO 4 J=1,I
    4 SM=SM+G(LI+J)*X(J)
    3 Y(I)=SM
      WRITE(61,23)
   23 FORMAT(3X*      Y(I)\      *)
      WRITE(61,22)(Y(I),I=1,N)
   22 FORMAT(3X,E16.7)
      CALL PLOT(Y,T,M,N)
      CALL CRODUR(Y,T,B,N,IDURN,ICDURN,PDSDRN,PDNDRN,
     QIDURT,ICDURT,PDSDRT,PDNDRT)
      GO TO 17
   19 STOP
      END
```


SUBROUTINE CRODUR CDC 6600 FTN V3.0-P295 OPT=1

```
      SUBROUTINE CRODUR(Y,T,B,N,IDURN,ICDURN,PDSDRN,PDNDRN,
     QIDURT,ICDURT,PDSDRT,PDNDRT)
      DIMENSION Y(100),IDURN(100),ICDURN(100),PDSDRN(100),PDNDRN(100),
     QIDURT(100),ICDURT(100),PDSDRT(100),PDNDRT(100)
      COMMON/MNP/FCT
      DELTA=0.01
      SUMK=0.
      SUMM=0.
      DO 40 J=1,100
      ICDURT(J)=0
   40 ICDURN(J)=0
      CALL CROSS(Y,B,K,N,IDURN,M,IDURT)
      SUMK=SUMK+K
      IF(K.EQ.0) GO TO 1
      DO 2 IDK=1,K
      L=IDURN(IDK)
    2 ICDURN(L)=ICDURN(L)+1
      WRITE(61,92)
   92 FORMAT(3X*      B                              SUMK        *)
    1 WRITE(61,51)B,SUMK
   51 FORMAT(3X,E12.3,6X,E16.7)
      IF(SUMK.LE.0.1) GO TO 95
      WRITE(61,30)
   30 FORMAT(1H1)
      WRITE(61,82)
   82 FORMAT(3X*  I        PDNDRN(I)      PDSDRN(I)  *)
      RSUMK=1./SUMK
      PDNDRN(1)=ICDURN(1)*RSUMK
      PDSDRN(1)=PDNDRN(1)
      SUMU=PDNDRN(1)
      SUMV=PDNDRN(1)
      I=2
    5 PDNDRN(I)=ICDURN(I)*RSUMK
      PDSDRN(I)=PDSDRN(I-1)+PDNDRN(I)
      SUMU=SUMU+I*PDNDRN(I)
      SUMV=SUMV+I*I*PDNDRN(I)
      IF(PDSDRN(I).GE.1.)GO TO 15
      IF(I.GE.100)GO TO 15
      I=I+1
      GO TO 5
   15 WRITE(61,11)(J,PDNDRN(J),PDSDRN(J),J=1,I)
   11 FORMAT(3X,I3,7X,F6.4,7X,F6.4)
      U=SUMU*DELTA
      V=SUMV*(DELTA**2)
      SIG=SQRT(V-U**2)
      WRITE(61,30)
      WRITE(61,90)
   90 FORMAT(3X,*      U              V              SIG  *)
      WRITE(61,91)U,V,SIG
   91 FORMAT(3X,F10.6,3X,F10.6,3X,F10.6)
   95 SUMM=SUMM+M
      IF(M.EQ.0) GO TO 71
      DO 72 IDM=1,M
      L=IDURT(IDM)
   72 ICDURT(L)=ICDURT(L)+1
```


```
  SUBROUTINE  CRODUR                                    CDC 6600 FTN V3.0-P296 OPT=1

                    WRITE(61,30)
                    WRITE(61,73)
                 73 FORMAT(3X*      B                          SUMM       *)
                 71 WRITE(61,74)B,SUMM
60               74 FORMAT(3X,E12.3,6X,E16.7)
                    IF(SUMM.LE.0.1) GO TO 96
                    WRITE(61,30)
                    WRITE(61,75)
                 75 FORMAT(3X*  I        PDNDRT(I)      PDSDRT(I)  *)
65                  RSUMM=1./SUMM
                    PDNDRT(1)=ICDURT(1)*RSUMM
                    PDSDRT(1)=PDNDRT(1)
                    SUTU=PDNDRT(1)
                    SUTV=PDNDRT(1)
70                  I=2
                 85 PDNDRT(I)=ICDURT(I)*RSUMM
                    PDSDRT(I)=PDSDRT(I-1)+PDNDRT(I)
                    SUTU=SUTU+I*PDNDRT(I)
                    SUTV=SUTV+I*I*PDNDRT(I)
75                  IF(PDSDRT(I).GE.1.)GO TO 25
                    IF(I.GE.100)GO TO 25
                    I=I+1
                    GO TO 85
                 25 WRITE(61,11)(J,PDNDRT(J),PDSDRT(J),J=1,I)
80                  UT=SUTU*DELTA
                    VT=SUTV*(DELTA**2)
                    SIT=SQRT(VT-UT**2)
                    WRITE(61,30)
                    WRITE(61,94)
85               94 FORMAT(3X,*      UT             VT          SIT  *)
                    WRITE(61,91)UT,VT,SIT
                 96 RETURN
                    END
```

YMIN
-.177E+01

YMAX
.283E+01

B	SUMK
.400E+00	.9000000E+01

I	PDNDRN(I)	PDSDRN(I)
1	.1111	.1111
2	.3333	.4444
3	.3333	.7778
4	0.0000	.7778
5	.1111	.8889
6	0.0000	.8889
7	0.0000	.8889
8	.1111	1.0000

U	V	SIG
.032222	.001433	.019876

B	SUMM
.400E+00	.1000000E+02

I	PDNDRT(I)	PDSDRT(I)
1	0.0000	0.0000
2	.2000	.2000
3	.1000	.3000
4	.2000	.5000
5	.1000	.6000
6	0.0000	.6000
7	0.0000	.6000
8	0.0000	.6000
9	0.0000	.6000
10	.1000	.7000
11	0.0000	.7000
12	0.0000	.7000
13	.2000	.9000
14	0.0000	.9000
15	.1000	1.0000

UT	VT	SIT
.071000	.007370	.048260

PROGRAM No. 25.

Name: STOCHY

Purpose: Repeat of program no. 24 with different limit. (Results only).

B	SUMK
0.	.8000000E+01

I	PDNDRN(I)	PDSDRN(I)
1	0.0000	0.0000
2	.1250	.1250
3	.2500	.3750
4	.2500	.6250
5	0.0000	.6250
6	0.0000	.6250
7	0.0000	.6250
8	.1250	.7500
9	.1250	.8750
10	0.0000	.8750
11	0.0000	.8750
12	.1250	1.0000

U	V	SIG
.056250	.004287	.033518

B	SUMM
0.	.9000000E+01

I	PDNDRT(I)	PDSDRT(I)
1	0.0000	0.0000
2	.1111	.1111
3	.2222	.3333
4	.1111	.4444
5	.2222	.6667
6	0.0000	.6667
7	0.0000	.6667
8	0.0000	.6667
9	.1111	.7778
10	0.0000	.7778
11	.1111	.8889
12	0.0000	.8889
13	0.0000	.8889
14	.1111	1.0000

UT	VT	SIT
.062222	.005400	.039095

PROGRAM No. 26

Name: CROSSY

Purpose: Program to determine the statistics of the excursions of a stochastic process above a certain limit.

```
PROGRAM    CROSSY                                        CDC 6600 FTN V3.0-
      PROGRAM CROSSY (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTP
      DIMENSION A(5050),G(5050),AM(100),X(100),Y(100),
     QIDURN(100),ICDURN(100),PDSDRN(100),PDNDRN(100),
     QIDURT(100),ICDURT(100),PDSDRT(100),PDNDRT(100)
      COMMON/MNP/FCT
      FCT=2.**(-48)
      READ(60,50)B
   50 FORMAT(F6.0)
      IF(B.LT.-0.1)STOP
      WRITE(61,30)
   30 FORMAT(1H1)
      N=100
      IDMN=5050
      BETA=0.1
      DELTA=0.01
      S=0.
      T=0.
      SP=UNI(1234567)
      DO 20 I=1,N
      DO 20 J=1,I
      K=(I*(I-1))/2+J
   20 A(K)=EXP(-BETA*(I-J)**2)
      DO 21 I=1,N
   21 AM(I)=0.0
      CALL DECOMR(A,G,N,IDMN)
      SUMK=0.
      SUMM=0.
      DO 40 J=1,100
      ICDURT(J)=0
   40 ICDURN(J)=0
      DO 90 IC=1,100
      DO 2 I=1,N
    2 X(I)=RNOR(0)
      DO 3 I=1,N
      SM=AM(I)
      LI=(I*(I-1))/2
      DO 4 J=1,I
    4 SM=SM+G(LI+J)*X(J)
      Y(I)=SM
      S=S+Y(I)
    3 T=T+(Y(I))**2
      CALL CROSS(Y,B,K,N,IDURN,M,IDURT)
      SUMK=SUMK+K
      SUMM=SUMM+M
      IF(K.EQ.0) GO TO 90
      IF(M.EQ.0) GO TO 90
      DO 22 IDK=1,K
      L=IDURN(IDK)
   22 ICDURN(L)=ICDURN(L)+1
      DO 72 IDM=1,M
      L=IDURT(IDM)
   72 ICDURT(L)=ICDURT(L)+1
   90 CONTINUE
      YAVE=S/(100*N)
      YVAR=T/(100*N)
```

```
      PROGRAM     CROSSY                                    CDC 6600 FTN V3.0-I
                  SIGY=SQRT(YVAR-YAVE**2)
                  WRITE(61,30)
                  WRITE(61,70)
                70 FORMAT(3X*       YAVE                YVAR              SIGY
      60          WRITE(61,71)YAVE,YVAR,SIGY
                71 FORMAT(3X,F10.4,3X,F10.4,3X,F10.4)
                  AVE=SUMK*0.01
                  WRITE(61,30)
                  WRITE(61,92)
      65        92 FORMAT(3X*       B                          AVE          *)
                  WRITE(61,51)B,AVE
                51 FORMAT(3X,E12.3,6X,E16.7)
                  IF(SUMK.LE.0.1) GO TO 95
                  WRITE(61,30)
      70          WRITE(61,82)
                82 FORMAT(3X*   I        PDNDRN(I)       PDSDRN(I)   *)
                  RSUMK=1./SUMK
                  PDNDRN(1)=ICDURN(1)*RSUMK
                  PDSDRN(1)=PDNDRN(1)
      75          SUMU=PDNDRN(1)
                  SUMV=PDNDRN(1)
                  I=2
                5 PDNDRN(I)=ICDURN(I)*RSUMK
                  PDSDRN(I)=PDSDRN(I-1)+PDNDRN(I)
      80          SUMU=SUMU+I*PDNDRN(I)
                  SUMV=SUMV+I*I*PDNDRN(I)
                  IF(PDSDRN(I).GE.0.9999)GO TO 15
                  IF(I.GE.100)GO TO 15
                  I=I+1
      85          GO TO 5
                15 WRITE(61,11)(J,PDNDRN(J),PDSDRN(J),J=1,I)
                11 FORMAT(3X,I3,7X,F6.4,7X,F6.4)
                  U=SUMU*DELTA
                  V=SUMV*(DELTA**2)
      90          SIG=SQRT(V-U**2)
                  WRITE(61,30)
                  WRITE(61,99)
                99 FORMAT(3X,*       U              V            SIG   *)
                  WRITE(61,91)U,V,SIG
      95        91 FORMAT(3X,F10.6,3X,F10.6,3X,F10.6)
                95 AVE=SUMM*0.01
                  WRITE(61,30)
                  WRITE(61,32)
                32 FORMAT(3X*       B                          AVE          *)
     100          WRITE(61,51)B,AVE
                  IF(SUMM.LE.0.1) GO TO 96
                  WRITE(61,30)
                  WRITE(61,75)
                75 FORMAT(3X*   I        PDNDRT(I)       PDSDRT(I)   *)
     105          RSUMM=1./SUMM
                  PDNDRT(1)=ICDURT(1)*RSUMM
                  PDSDRT(1)=PDNDRT(1)
                  SUTU=PDNDRT(1)
                  SUTV=PDNDRT(1)
     110          I=2
```

```
      PROGRAM      CROSSY                                    CDC 6600 FTN V3.0-I
                     85 PDNDRT(I)=ICOURT(I)*RSUMM
                        PDSDRT(I)=PDSDRT(I-1)+PDNDRT(I)
                        SUTU=SUTU+I*PDNDRT(I)
                        SUTV=SUTV+I*I*PDNDRT(I)
  115                   IF(PDSDRT(I).GE.0.9999)GO TO 25
                        IF(I.GE.100)GO TO 25
                        I=I+1
                        GO TO 85
                     25 WRITE(61,11)(J,PDNDRT(J),PDSDRT(J),J=1,I)
  120                   UT=SUTU*DELTA
                        VT=SUTV*(DELTA**2)
                        SIT=SQRT(VT-UT**2)
                        WRITE(61,30)
                        WRITE(61,94)
  125                94 FORMAT(3X,*      UT            VT          SIT   *)
                        WRITE(61,91)UT,VT,SIT
                     96 STOP
                        END
```

YAVE	YVAR	SIGY
.0212	1.0121	1.0058

B	AVE
0.	.7450000E+01

I	PDNDRN(I)	PDSDRN(I)
1	.0591	.0591
2	.1087	.1678
3	.1168	.2846
4	.1450	.4295
5	.1074	.5369
6	.0792	.6161
7	.0819	.6980
8	.0537	.7517
9	.0389	.7906
10	.0322	.8228
11	.0322	.8550
12	.0282	.8832
13	.0188	.9020
14	.0107	.9128
15	.0121	.9248
16	.0067	.9315
17	.0174	.9490
18	.0067	.9557
19	.0107	.9664
20	.0027	.9691
21	.0067	.9758
22	.0027	.9785
23	.0013	.9799
24	.0027	.9826
25	.0067	.9893
26	.0027	.9919
27	0.0000	.9919
28	.0040	.9960
29	0.0000	.9960
30	.0013	.9973
31	0.0000	.9973
32	.0013	.9987
33	0.0000	.9987
34	.0013	1.0000

U	V	SIG
.067195	.007288	.052659

I	PDNDRT(I)	PDSDRT(I)
1	.0694	.0694
2	.0972	.1667
3	.1250	.2917
4	.1136	.4053
5	.0985	.5038
6	.0795	.5833
7	.0619	.6452
8	.0556	.7008
9	.0417	.7424
10	.0303	.7727
11	.0265	.7992
12	.0215	.8207
13	.0189	.8396
14	.0227	.8624
15	.0076	.8699
16	.0114	.8813
17	.0088	.8902
18	.0051	.8952
19	.0101	.9053
20	.0013	.9066
21	.0076	.9141
22	.0051	.9192
23	.0013	.9205
24	.0038	.9242
25	.0025	.9268
26	.0025	.9293
27	0.0000	.9293
28	.0013	.9306
29	.0013	.9318
30	0.0000	.9318
31	0.0000	.9318
32	.0013	.9331
33	0.0000	.9331
34	0.0000	.9331
35	0.0000	.9331
36	0.0000	.9331
37	.0013	.9343
38	0.0000	.9343
39	0.0000	.9343
40	0.0000	.9343
41	0.0000	.9343
42	0.0000	.9343
43	0.0000	.9343
44	.0013	.9356
45	0.0000	.9356
46	0.0000	.9356
47	0.0000	.9356
48	.0013	.9369
49	0.0000	.9369
50	0.0000	.9369

B	AVF
0.	.7920000E+01

UT	VT	SIT
.062803	.007098	.056156

PROGRAM No. 27

Name: CROSSY

Program: Program No. 26 modified to correct error and to make general.

```
PROGRAM     CROSSY                                    CDC 6600 FTN V3.0-P

      PROGRAM CROSSY (INPUT,OUTPUT,TAPE60=INPUT,TAPE61=OUTPU
      DIMENSION A(5050),G(5050),AM(100),X(100),Y(100),
     QIDURN(100),ICDURN(100),PDSDRN(100),PDNDRN(100),
     QIDURT(100),ICDURT(100),PDSDRT(100),PDNDRT(100)
 5    COMMON/MNP/FCT
      FCT=2.**(-48)
      READ(60,50)B
   50 FORMAT(F6.0)
      IF(B.LT.-0.1)STOP
10    WRITE(61,30)
   30 FORMAT(1H1)
      N=100
      NN=100
      IDMN=5050
15    BETA=0.05
      DELTA=1./N
      S=0.
      T=0.
      ST=UNI(1234567)
20    DO 20 I=1,N
      DO 20 J=1,I
      K=(I*(I-1))/2+J
   20 A(K)=EXP(-BETA*(I-J)**2)
      DO 21 I=1,N
25    21 AM(I)=0.0
      CALL DECOMP(A,G,N,IDMN)
      SUMK=0.
      SUMM=0.
      DO 40 J=1,N
30    ICDURT(J)=0
   40 ICDURN(J)=0
      DO 90 IC=1,NN
      DO 2 I=1,N
    2 X(I)=RNOR(0)
35    DO 3 I=1,N
      SM=AM(I)
      LI=(I*(I-1))/2
      DO 4 J=1,I
    4 SM=SM+G(LI+J)*X(J)
40    Y(I)=SM
      S=S+Y(I)
    3 T=T+(Y(I))**2
      CALL CROSS(Y,B,K,N,IDURN,M,IDURT)
      SUMK=SUMK+K
45    SUMM=SUMM+M
      IF(K.EQ.0) GO TO 90
      IF(M.EQ.0) GO TO 90
      DO 22 IDK=1,K
      L=IDURN(IDK)
50    22 ICDURN(L)=ICDURN(L)+1
      DO 72 IDM=1,M
      L=IDURT(IDM)
   72 ICDURT(L)=ICDURT(L)+1
   90 CONTINUE
55    YAVE=S/(N*NN)
```

PROGRAM CROSSY CDC 6600 FTN V3.0-P

```
      YVAR=T/(N*NN)
      SIGY=SQRT(YVAR-YAVE**2)
      WRITE(61,30)
      WRITE(61,70)
   70 FORMAT(3X*       YAVE                YVAR                SIGY  *
      WRITE(61,71)YAVE,YVAR,SIGY
   71 FORMAT(3X,F10.4,3X,F10.4,3X,F10.4)
      AVE=SUMK/NN
      WRITE(61,30)
      WRITE(61,92)
   92 FORMAT(3X*       B                                AVE         *)
      WRITE(61,51)B,AVE
   51 FORMAT(3X,E12.3,6X,E16.7)
      IF(SUMK.LE.0.1) GO TO 95
      WRITE(61,30)
      WRITE(61,82)
   82 FORMAT(3X*   I        PDNDRN(I)      PDSDRN(I)   *)
      RSUMK=1./SUMK
      PDNDRN(1)=ICDURN(1)*RSUMK
      PDSDRN(1)=PDNDRN(1)
      SUMU=PDNDRN(1)
      SUMV=PDNDRN(1)
      I=2
    5 PDNDRN(I)=ICDURN(I)*RSUMK
      PDSDRN(I)=PDSDRN(I-1)+PDNDRN(I)
      SUMU=SUMU+I*PDNDRN(I)
      SUMV=SUMV+I*I*PDNDRN(I)
      IF(PDSDRN(I).GE.0.9999)GO TO 15
      IF(I.GE.N)GO TO 15
      I=I+1
      GO TO 5
   15 WRITE(61,11)(J,PDNDRN(J),PDSDRN(J),J=1,I)
   11 FORMAT(3X,I3,7X,F6.4,7X,F6.4)
      U=SUMU*DELTA
      V=SUMV*(DELTA**2)
      SIG=SQRT(V-U**2)
      WRITE(61,30)
      WRITE(61,99)
   99 FORMAT(3X,*      U                V           SIG   *)
      WRITE(61,91)U,V,SIG
   91 FORMAT(3X,F10.6,3X,F10.6,3X,F10.6)
   95 AVF=SUMM/NN
      WRITE(61,30)
      WRITE(61,32)
   32 FORMAT(3X*       B                                AVF         *)
      WRITE(61,51)B,AVF
      IF(SUMM.LE.0.1) GO TO 96
      WRITE(61,30)
      WRITE(61,75)
   75 FORMAT(3X*   I        PDNDRT(I)      PDSDRT(I)   *)
      RSUMM=1./SUMM
      PDNDRT(1)=ICDURT(1)*RSUMM
      PDSDRT(1)=PDNDRT(1)
      SUTU=PDNDRT(1)
      SUTV=PDNDRT(1)
```

```
PROGRAM      CROSSY                                        CDC 6600 FTN V3.0-P

                    I=2
                85, PDNDRT(I)=ICDURT(I)*RSUMM
                    PDSDRT(I)=PDSDRT(I-1)+PDNDRT(I)
                    SUTU=SUTU+I*PDNDRT(I)
115                 SUTV=SUTV+I*I*PDNDRT(I)
                    IF(PDSDRT(I).GE.0.9999)GO TO 25
                    IF(I.GE.N)GO TO 25
                    I=I+1
                    GO TO 85
120             25  WRITE(61,11)(J,PDNDRT(J),PDSDRT(J),J=1,I)
                    UT=SUTU*DELTA
                    VT=SUTV*(DELTA**2)
                    SIT=SQRT(VT-UT**2)
                    WRITE(61,30)
125                 WRITE(61,94)
                94  FORMAT(3X,*      UT              VT          SIT  *)
                    WRITE(61,91)UT,VT,SIT
                96  STOP
                    END


   SUBROUTINE  CROSS

                    SUBROUTINE CROSS(Y,B,K,N,IDURN,M,IDURT)
                    DIMENSION Y(N),IDURN(N),IDURT(N)
                    K=0
                    M=0
 5                  I=1
                    L=I
                    IF(Y(I).GE.B)GO TO 5
                1   I=I+1
                    IF(I.GT.N)GO TO 12
10                  IF((Y(I).GE.B).AND.(Y(I-1).LT.B))GOTO3
                    GOTO1
                5   K=K+1
                    J=I
                    GO TO 4
15              3   K=K+1
                    M=M+1
                    J=I
                    IDURT(M)=(J-L)
                4   I=I+1
20                  L=I
                    IF(I.GT.N)RETURN
                    IDURN(K)=(I-J)
                    IF(Y(I).LT.B)GOTO1
                    GOTO4
25              12  M=M+1
                    IDURT(M)=(N-L+1)
                    RETURN
                    END
```